공학박사학위논문

Markerless Reconstruction of Human Motion Data

사람 동작의 마커없는 재구성

2017년 2월

서울대학교 대학원
전기.컴퓨터공학부
양 경 용

Markerless Reconstruction of Human Motion Data

사람 동작의 마커없는 재구성

지도교수 이 제 희

이 논문을 공학 박사 학위논문으로 제출함

2016년 10월

서울대학교 대학원

전기.컴퓨터공학부

양 경 용

양경용의 공학박사 학위논문을 인준함

2016년 12월

위 원 장	김 명 수
부위원장	이 제 희
위 원	유 승 주
위 원	김 건 희
위 원	권 태 수

Abstract

Markerless human pose recognition using a single-depth camera plays an important role in interactive graphics applications and user interface design. Recent pose recognition algorithms have adopted machine learning techniques, utilizing a large collection of motion capture data. The effectiveness of the algorithms is greatly influenced by the diversity and variability of training data. Many applications have been developed to use human body as a controller to utilize these pose recognition systems. In many cases, using general props help us perform immersion control of the system. Nevertheless, the human pose and prop recognition system is not yet sufficiently powerful. Moreover, there is a problem such as invisible parts lower the quality of human pose estimation from a single depth camera due to an absence of observed data.

In this thesis, we present techniques to manipulate the human motion data for enabling to estimate human pose from a single depth camera. First, we developed method that resamples a collection of human motion data to improve the pose variability and achieve an arbitrary size and level of density in the space of human poses. The space of human poses is high-dimensional and thus brute-force uniform sampling is intractable. We exploit dimensionality reduction and locally stratified sampling to generate either uniform or application-specifically biased distributions in the space of human poses. Our algorithm is learned to recognize such challenging poses such as sit, kneel, stretching and yoga using a remarkably small amount of training data. The

recognition algorithm can also be steered to maximize its performance for a specific domain of human poses. We demonstrate that our algorithm performs much better than Kinect SDK for recognizing challenging acrobatic poses, while performing comparably for easy upright standing poses. Second, we find out environmental object which interact with human beings. We proposed a new props recognition system, which can applied on the existing human pose estimation algorithm, and enable to powerful props estimation with human poses at the same times. Our work is widely applicable to various types of controllers system, which deals with the human pose and addition items simultaneously. Finally, we enhance the pose estimation result. All the part of human body cannot be always estimated from the single depth image. In some case, some body parts are occluded by other body parts, and sometimes estimation system fail to success. For solving this problem, we construct novel neural network model which called autoencoder. It is constructed from huge natural pose data. Then it can reconstruct the missing parameter of human pose joint as new correct joint. It can be applied to many different human pose estimation systems to improve their performance.

keywords: Computer Graphics, Character Animation, Motion Capture, Human Pose Recognition, Uniform Sampling, Machine Learning, Deep Learning.

Student Number: 2008-30879

Contents

List of Figures

Chapter 1

Introduction

It is the one of final goals of human science that understanding, analyzing, reenacting and applying the human beings. Among the many parts of human, research which focuses the human movement has been thought an important work. In the 19 century, Eadweard Muybridge first start records the movements of animals and human using sequential photography (See Figure 1.1. It is a great worth. These results have been the basis of various area researchers, for example, biomechanics and kinesiology.

Like this, many researchers use sequential image information for analyzing the human movements and applying these results. Before the technological developments, people performed a fast sketch which is an available technique for record human movements. As technology develops, photography becomes a major technique for recording human movements. However, there is only planar visual information using photography, so we must deal with additional human movement data on the 3D space.

As computer development, much work of the real worlds has become in many cases to be processed transferred to a computer environment. Many people have

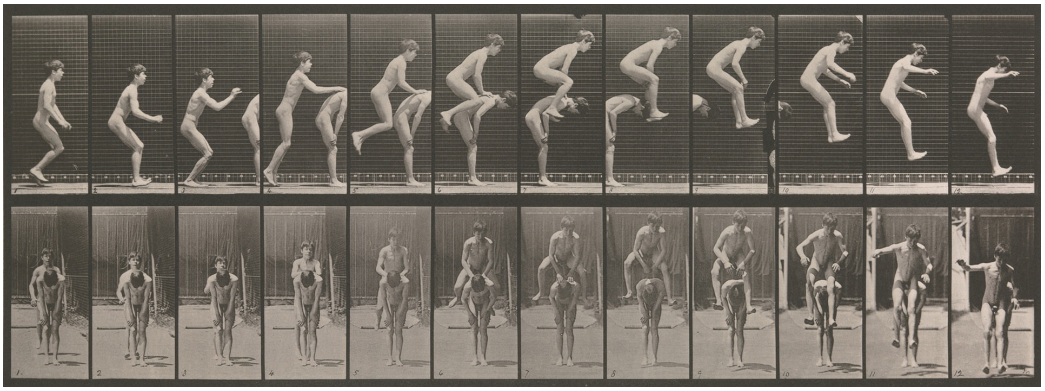


Figure 1.1: *Pioneer photography of human motion (from Boys Playing Leap Frog, Eadweard Muybridge, 1887.)*

been interested in the various types of data can be understood by the computer form of the real world. For doing this, people have been much effort to understand movement of the human. Among them capturing human pose and motion data is the most important issues, many researchers have been perform solve this problem. For the computer understands the human movements, we define the human pose in 3D space and express the human movements with time series set. This sequential work is developed, which called motion capture and is widely used as common tools. Now a days optical motion capture system is established and it can provide high quality human motion data to academic, movie, game and other various purpose. There has been a breakthrough in each field and motion capture system. However there are still somethings to improve on establishing high-cost equipment and labor-intensive and so on.

Recently it has been made many studies to make can convert exactly human posture and motion information to the computer environment without high-cost optical motion capture. These can be used freely transferred information to a computer en-

vironment anywhere and with only a simple apparatus for obtaining the human pose and motion. Then it allow any person who has a simple computer system can capture his/her pose and motion easily and the possibility of use is intended to be infinite.

Among the various motion capture methods, the markerless motion capture has recently attracted attention. Unlike the conventional method, this method can perform capturing to transfer human movements information to a computer environment without wearing additional equipment. This can save the time and effort required to wear additional equipment, and can prevent unnaturalness and behavioral limitations when wearing the equipment. It is possible to capture at a fairly accurate level using multiple pre-calibrated cameras in the studio.

Especially, the method of recognizing and capturing the human posture using only a single single depth camera is very important meaning. Until now, motion capture has been a means used by producers. In the future, users will be able to capture their own actions and use them to consume content. While existing controller only inputs pre-defined command set as a input, markerless capture controller accepts all the human analog operations as inputs and can be used as a variety of controls. This not only lowers the entry barriers to motion capture, but it also makes sense for motion capture attempts to be easily made available to all personal users.

An additional process must be applied to recognize the human pose without using a marker that directly represents the human pose. Among them, in order to obtain the human pose from the depth image as an input, a model capable of distinguishing the human pose from the single depth image through the pre-learning should be learned. This process uses pre-distinguished human poses as learning data. However, it is

difficult for a person to actually use various poses precisely and use them as learning data. When synthetic learning data is used, time required to acquire data can be shortened, and necessary pose can be easily obtained. In the process of obtaining such synthetic learning data, the pose data obtained by optical motion capture will be used. This is because, as mentioned earlier, it takes much time, but pose data with high accuracy can be obtained.

We are facing following issues:

- Can you generate training data to ensure performance of markerless pose recognition model?
- Is it possible to apply existing learned model to human pose with other objects at the same time for recognition?
- If the pose you find is not complete, how will you solve it?

In this thesis, we present a solution to these issues and complete the markerless pose recognition system using a single depth camera. We want to be able to recognize this human poses accurately and effectively by learning prior knowledge which based on understanding human movement to the computer. Contribution of this work is following.

Controllable Data Sampling in the Space of Human Poses.

It is a much difficult to converting real human movement data into computer environment. For acquiring enough through this process, we need somewhat repetition activity. If an activity is easy to acting, there is no problem. But if an activity is hard to acting, we need huge time and cost. In the view of a pose domain, these

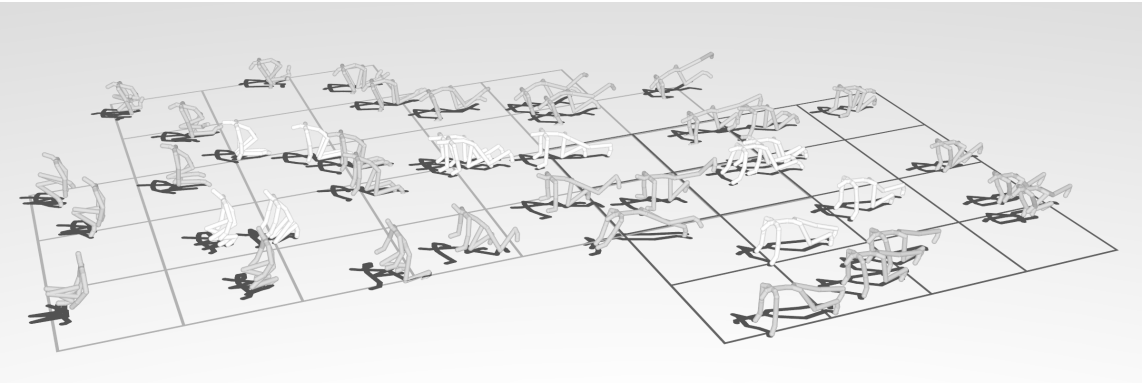


Figure 1.2: *Illustration of human pose uniform sampling*

unsatisfactory are more serious. Especially, a pose which is somewhat concealed by other body parts is hard for getting from the motion capture system. If do so, more expensive time needed. As a result, generating these kinds of data successfully based on existing data, will be an awesome meaningful work.

Based on the given human movement data, we need to understand a property of human pose data, for generating various style and detail pose with maintaining the contents of data. Because dimension of human pose is much high, it is not easy to apply common generating algorithm. Especially brute-force sampling never work in this problem. Therefore, we must understand the space of human poses which forms a manifold. Based on these knowledge we divide the space and reduce the dimension with respect to property of human pose data then we can generate a desirable new data.

We have propose a method that allows it to be possible to improve the performance of all the solutions to leverage training data by utilizing the full advantage of existing motion capture data. This will be apply on the process of choosing training data which is used for system learning. A lot of motion capture data already accessible

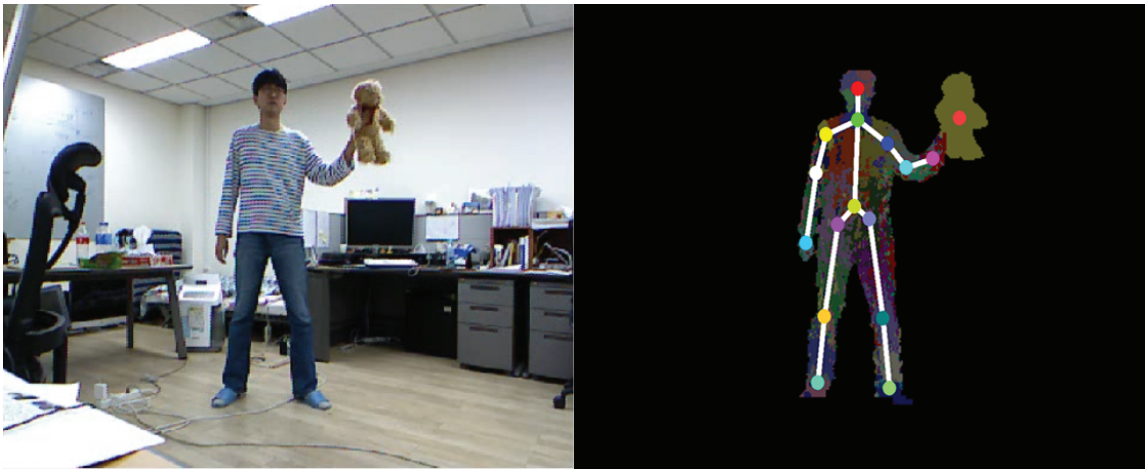


Figure 1.3: *Illustraion of human pose and prop estimation*

on public motion database by web, but these data are biased. There are enough basic and standing motion, but highly dynamic and special contents motion are not. There is also needed for even more effort to get himself these data. The performance of most pose recognition algorithm using prior learning is greatly affected by the distribution of the training data. Therefore it is possible to improve the posture recognition performance using uniform reconstructed data in the human pose space rather than using the learning data exists these imbalances.

Human pose estimation with interacting prop from single depth image. For second issues, we divide human poses and additional arbitrary props. The pose estimation method based on the learned model operates only when the human posture enters the input. If an object and a person other than a person come together and want to recognize it, you have to worry about it. If other objects and human come in as inputs together and want to recognize them, we have to consider that situation. Generally, if a human pose is to be recognized using prior learning, it is assumed that

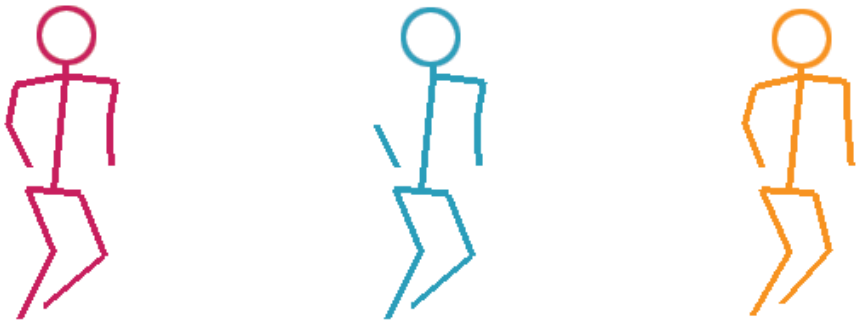


Figure 1.4: *Illustration of missing joint estimation*

data other than a human pose is considered as noise in the surrounding environment, and only human data is used as input data. The reason is that the pose structure of a human is almost the same, everyone is different in volume and length. However, if other object is added as an input, the learning should be performed with the environment. In order to learn this properly, it is necessary to use the input object together with the human pose as the learning data. However, the learning process generally uses a large amount of data, and therefore takes a long time to perform. Therefore, it is necessary to use the input of objects other than human pose separately, and a method to solve this problem efficiently in the process of pose recognition is needed. We propose a method to easily distinguish human and props from each other during pose recognition process.

Enhance the estimation of human pose from incomplete joints.

In the markerless motion capture system, pre-learning is performed based on depth image information, which is input data. Sometimes, however, estimated human pose information with this learning model shows incomplete results. For example, the learn-

ing model may not be able to labeling enough information about a single joint to be found at all, and the joint position may show a slightly out-of-place position. Many researchers try to solve this problem by using tracking, which uses data before and after. We still use a single depth image to successfully estimate the joint that failed to estimate. To do this, we want to use the position of the estimated joint result rather than the depth image. We will generate the model of pose structure using joint positions of natural poses and improve pose estimation performance using this learning model. We proposed a method to generate a neural network model that outputs complete posture without any additional information when incomplete posture information is given as input using joint position

Our thesis is structured as follows. Chapter 2 introduces background and related work about the improvement the human pose data and identify the movement mechanism. In Chapter 3 illustrate markerless recognition system and preprocess parts of our thesis. Chapter 4 describes a new method that resamples a collection of human motion data to improve pose variability. In Chapter 5 we present how to recognize human pose and arbitrary props simultaneously. Chapter 6 explain how to recover the missing joints of human pose and to improve performance. Finally, Chapter 7 concludes thesis and describe the future work.

Chapter 2

Background

This chapter gives the background knowledge and history of related work that presented in the thesis.

2.1 Research on Motion Data

It has been performed use motion capture data in equipped facility for a long time. In this place, which is called motion capture studio, human act various actions and capture them. However it is much difficult to work. This process is time and labor intensive. Even there is no guarantee get desirable data exactly. Therefore researchers want to use these captured data effectively, they develop motion graph [38, 34, 3, 50]. Using motion graph, we can generate various motion sequence continuously with a limited number of motion data. This structure used for understanding and researching human movement frequently. There have also been studies that utilize motion data by parameterizing [33], blending [49], classifying and synthesizing motions [35].

2.2 Human Pose Estimation

Estimation of human body poses from images has been a major goal of computer vision. The use of a depth camera greatly simplifies the human pose estimation problem. Shotton et al. [58] developed a body part recognition algorithm that is part of the commercial Kinect™ system. The algorithm first segments different body parts using random decision trees and then estimates joint positions from body part labels. Girshick et al. [17] suggested an alternative regression-based method that estimates joint positions directly from depth images without intermediate steps for estimating body part labels. Sun et al. [61] employed conditional regression forests to incorporate prior knowledge and global variables, such as the user's height and limb lengths, to improve the recognition performance. Alternatively, Ye et al. [70] and Baak et al. [4] independently explored a data-driven approach, that explicitly maintains a database of human poses and searches best matching poses at runtime to facilitate pose reconstruction. Wei et al. [67] combined full-body tracking with body part recognition to improve the robustness of the algorithm. For improving joint estimation time, Jung et al. [29] trained the probability distribution of the direction toward joint from pixels.

The Kinect™ cameras have stimulated follow-up studies and have been employed in a variety of user interfaces and applications. The use of a depth camera enables the tracking of full articulation of human hands [48] and hand pose recognition [51, 31], and facial expression recognition [68]. For more robust pose reconstruction from the self-occlusion and noises, Liu et al. [43] adopted the Gaussian Process model. Low-cost, real-time, 3D reconstruction of the environment using a hand-held moving depth camera has been explored [27]. Touch-free, gesture interfaces are attracting attention

in medical applications because of the sterilization requirements in the operating room [15].

There are some trials to reconstruct the human motions from hand-held sensors which are without complex set up of human body [19, 32]. Other research reconstruct human motion of kinematics and dynamics data using depth cameras and sensors together [71]. By adding a simple infrared module to the 2D camera, depth sensing and near-object capture have become possible [54]. Recently, combining RGB and depth images show good performance on 3D human motion capture [11]. A nonrigid reconstruction system which is satisfying to respatio-temporally coherent has developed from multiple RGBD cameras [10]. Rhodin et al [52] made it to possible egocentric full-body motion capture regardless to the environments using fisheye view with a convolution network body-part detector.

Nguyen et al. [47] show a technique for interacting with a physical model. There has been an attempt to train human poses and props together for recognizing [20], though it is dependent on the specific viewport of human pose and props.

2.3 Machine Learning on Human Pose Estimation

Human pose estimation/tracking algorithms are often learned from a large collection of training data. Density estimation of human pose data [6] uncovers the nonlinear structure of the data, which in turn can be exploited for pose estimation and tracking. Our goal is different from density estimation of human pose data. Biased training data affect the learning performance. Yanmada et al. [69] applied a weighted regression method to eliminate these biased in training data sets. Numerous

approaches to change the data distribution from known data density are available. Among the possibilities, we adopt the PCA based method for verifying that our algorithm works well despite being a basic and simple methods based on dimension reduction techniques. We explicitly change the distribution of the training data to make it more effective for the learning process.

2.4 Dimension Reduction and Uniform Sampling

The key challenge is coping with the size and high-dimensionality of the training data. Lau et al. [36] explored the modeling of spatial and temporal variations in motion data based on a dynamic Baysian network model, which takes a small number of motion examples as input and produces their variants. The results were demonstrated with less than ten examples. A Gaussian Process Latent Variable Model is a good approach among non-linear probabilistic PCA techniques [37]. But in only showed fine results on a somewhat small number of examples. It is necessary to synthesize a uniform sampling of human poses from a much larger (typically, ten of thousands to millions) set of example poses. For motion data, it has been work dimensionally reduced representation for efficient motion retrieval [14] and synthesize the motion in low-dimensional spaces [56]

The notion of uniform sampling has been explored in the context of Poisson disk sampling and blue noise [9, 12]. A number of sophisticated algorithms for Poisson disk sampling, have been reported, but they do not generalize easily to deal with high-dimensional data. Alternatively, the training data can be projected into lower-dimensional space [57, 18]. Most dimensionality reduction algorithms require $O(n^3)$

computation and $O(n^2)$ memory, which is not feasible with a large training set. Exploiting locality is a common approach in large-scale machine learning [53].

2.5 Neural Networks on Motion Data

Recently, a neural network show a remarkable performance in vision area. It show a high quality results on video a classification [30] and facial recognition [46]. Moreover, applying motion data is successfully used for action recognition [28], action tracking [13] and speech recognition [2]. At human motion data, it is possible to predict next pose from several previous frame [62]. Tompson et al. [63] applied a convolutional neural network to feature extraction for enabling continuous tracking of the skinned hand model. Wang et al. [66] encode motion data into deep signature which is small and condensed representation. They can perform effectively motion compression, indexing, retrieval and reconstruction. It has been trying to learning a manifold of human motion [23] and can synthesize character movements with respect to the manifold of human motion [22]. Bogo et al. [5] proposed a method to find out the human pose and shape from a single image using CNN-based training and statistical body shape model. Hong et al. [24] performed a successful pose reconstruction by mapping two autoencoder each of them can represent 2D silhouettes and 3D poses.

Chapter 3

Markerless Human Pose Recognition System

3.1 System Overview

In this section, we will talk about a system that constitutes a markerless human pose recognition system proposed by Shotton et al [58]. An overview of markerless recognition process and our thesis is illustrated in Figure 3.1. Markerless motion capture using a single depth camera proceeds as follows. If a person takes a pose in front of a depth camera, the corresponding depth value is input to the camera. The camera obtains a depth image, then labels it by predicting which one of the pixels corresponds to each human body part using a pre-learned learning model. Finally human pose is estimated using the these labeled pixel data. We will explain separately this process into three parts: the preprocessing data process, the randomized decision tree, and the pose estimation evaluation process.

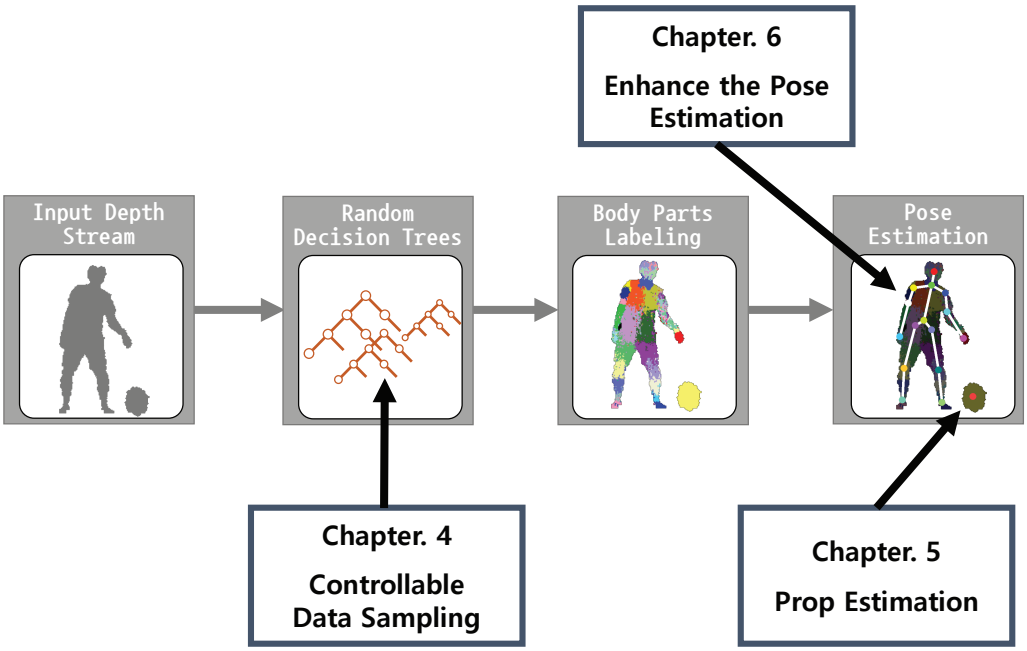


Figure 3.1: *Thesis overview*

3.2 Preprocessing Data Process

Processing Motion Data

As mentioned in chapter 1, there is the way to obtain the learning data by directly acquiring the actual human poses. However, this method is extremely time consuming and labor intensive, and it may be more difficult to obtain desired poses. Therefore, we have generated synthetic learning data using motion capture data.

We conducted experiments using 150 minutes of motion data downloaded from motion databases available on the web [8, 59]. The motion data recorded a variety of human activities including locomotion, gesture, dance, martial arts, acrobatic performance, yoga, stretching, sports, and so on. The sampling rate of motion capture data is usually higher than required for our purpose. We subsampled motion data to

maintain three frames per second in our data set. The poses in the data set are dominantly in an upright stance because such poses are easy to record in motion capture. On the other hand, acrobatic poses are not as abundant as standing poses in the public motion databases. We classified individual frames of motion data into three categories (see Figure 3.2). The classification is based on the difficulty of recognition in computer algorithms.

- **Type I (Upright Stand)** A Type I pose has the body upright standing on the feet and has no contact between the upper and lower body parts.
- **Type II (Acrobatic Stand)** A Type II pose has the upper body leaning more than 45° from the vertical axis, or either the knee or the foot above the height of the pelvis, or has any of the upper body part and its lower body part in contact (e.g., a hand on a knee).
- **Type III (Sit and Squat)** A Type III pose has the height of the pelvis from the ground lower than the knee height in an upright position, or has a body part other than the feet in contact with the ground surface.

Type I poses are abundant in the training data and thus pose recognition algorithms work well with Type I poses, whereas we do not have sufficient Type II and Type III data to learn a reliable recognition algorithm. In particular, Type III poses are very difficult to recognize.

The articulated figure has 20 body parts and 19 joints (see Figure 3.3). The pose of the figure is represented as a heterogeneous array $(\mathbf{p}_0, \mathbf{q}_0, \cdots, \mathbf{q}_{19})$, where $\mathbf{p}_0 \in R^3$ and $\mathbf{q}_0 \in S^3$ are the position and orientation of the root segment (pelvis)

Figure 3.2: *Types of human poses. Type I (left four images) includes upright standing poses. Type II (middle four images) includes acrobatic standing poses. Type III (right four images) includes human poses sitting, squatting, and lying down on the ground.*

and $\mathbf{q}_i \in S^3$ for $i > 0$ is a unit quaternion representing the configuration of the i-th joint. Given a collection of pose data, their position and orientation should be normalized to remove the translation in the horizontal plane and the rotation about the vertical axis. We use the optimal distance metric for articulated poses to convert pose data into normalized pose vectors in R^{60} [40]. The skin model is a polygonal mesh with a texture image, which encodes body parts in different colors. The articulated skeleton is embedded in the skin model and thus the skin model deforms driven by the skeleton. Rendering of the skin model with depth information at each pixel generates a collection of synthetic depth images, which serve as training data to learn the body part recognition algorithm.

Create 3D Human Mesh Models and Rigging

The data that we want to obtain is not just motion data that is generally used to express human motion, but rather the data that is input to the depth camera. Therefore, we should obtain the data by making the collected poses as if the real human took it in front of the depth camera This requires a 3D mesh model that can represent the actual human volume which is similar to real human. We have created several 3D mesh models that are similar to common style people based on the standard body shape information of a human. Because the recognition algorithm is a scale invariant, we try to create various models of height-to-weight ratio. The details are as shown in figure 3.4.

In order to rig the previously collected motion data and generated models, we used a Motion builder which is commercial tool. [25]

Synthesize depth image

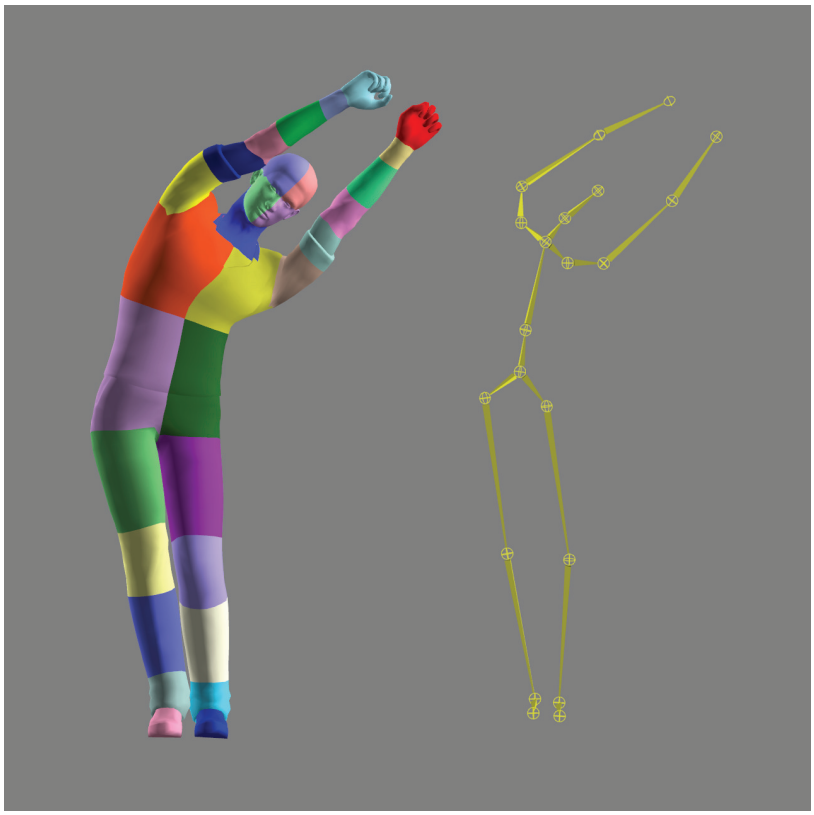


Figure 3.3: *The geometric model and its skeleton*

Setting the depth camera in the same state as actual person standing in front of the depth camera and getting the depth image. (See Figure 3.5) The position of the depth camera is determined in order R_y, Z, Y, R_x. The depth camera always looks at the origin, where the virtual 3D human model stand, and change the view direction according to R_y. Z is the distance from subject to depth camera, Y is the height of depth camera and R_x is the tilt of depth camera. We can generate 320x240 size synthetic depth images which includes depth value and body parts label.

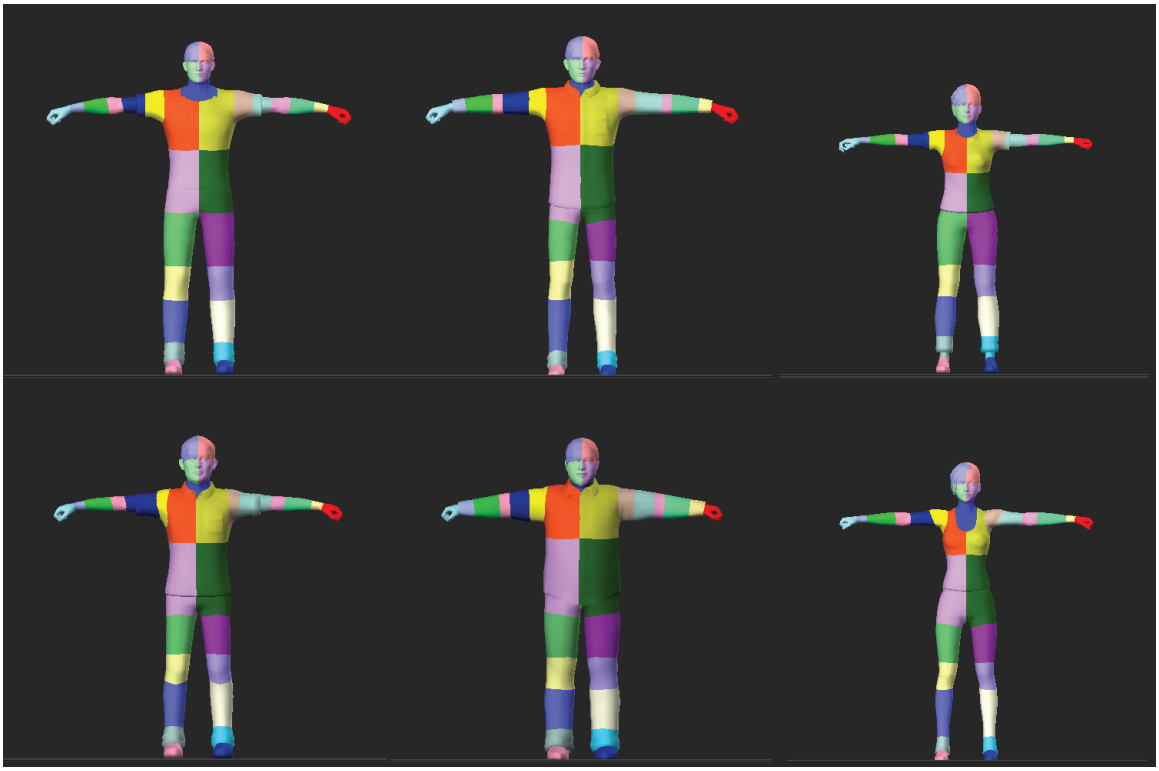


Figure 3.4: *3D human model (Top) man 185cm 70kg, man 188cm 75kg, woman 158cm 49kg (Botton) man 179cm 73kg, man 178 100kg, woman 169cm 51kg*

3.3 Randomized Decision Tree

The random decision trees are learned from a large collection of synthetic depth images. We generated synthetic depth images by rendering human body models, which have textures to label individual body parts (see Figure 3.3). Motion data are retargeted to each individual body model to animate, and rendered at different viewpoints to generate labeled depth images. The synthetic depth images represent the variations in body shapes, full-body poses, and viewing directions. The random decision trees would be resilient to such variations if the synthetic depth images provided sufficient variability. Among the aforementioned three categories, achieving variation in full-

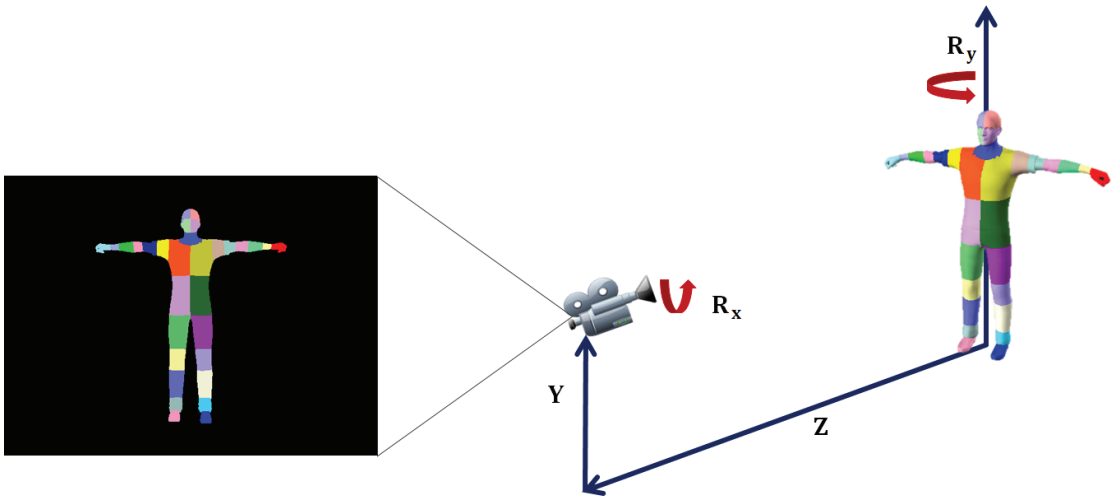


Figure 3.5: *Synthetic depth generation*

body poses is the most challenging. Our main contribution is a motion resampling algorithm that improves the pose distribution in the training data set.

The property of a Random Decision Tree is close to a non-parametric model such as k-nearest neighbor. The distribution of the learning model thus generally does not affect the performance. But the pose data that are obtained from a public database have too much sparsity and empty regions in the space of human poses because of the vastness of the space. They consequently give unadoptable classification results. From the view of machine learning, our work effectively generates adequate models in the learning space for successful learning.

At runtime, random decision trees take a stream of depth images from a depth camera and decide automatically which body part each pixel belongs to. The 3D joint positions are estimated from the body-part-labeled depth images.

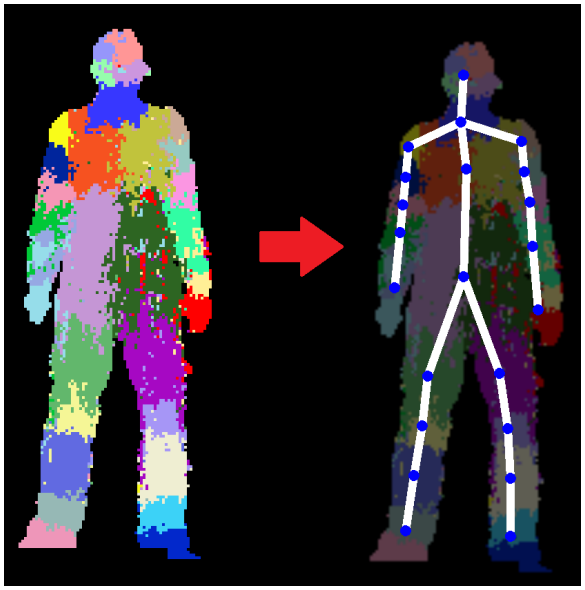


Figure 3.6: *Body parts labeling to joint estimation*

3.4 Joint Estimation Process

After depth pixel labeling process, the pose joint estimation process is performed. We have estimated the joint pose to be as accurate as possible in real-time using a single depth image. Basically we estimate one joint from one label, exceptionally hand and wrist, foot and ankle are considered as one joint. Moreover head joint is estimated from 4 head labels, upper body and lower body is estimated from each 2 body labels. Consequently, we estimate 22 temporary joint positions from 31 body parts labeling results(see figure 3.6). Each pixel from the labeling result is converted to a 3D point from the 2D point position and the depth value of the pixel. Due to depth camera specification, the p_x value is from 0 to 320, p_y value is from 0 to 240, and the depth value is from 400 to 4,000. The 3D point position $\mathbf{P} = (P_x, P_y, P_z)$ is determined as follows.

$$P_x = \frac{(160-p_x)}{Ratio} * depth \; P_y = \frac{(120-p_y)}{Ratio} * depth \; P_z = depth$$

$\frac{1}{Ratio}$ is the normalization term for depth scale invariance.

First, we extracted the candidates for each label. For perform this process, we uses the transformed 3D points and the mean shift clustering[7]. In this case, the radius value R is differently applied for the mean shift for each label L_i.

$$R(L_i) = k \cdot \sqrt{(max(N(L_i), \tau_r)}$$

k is a constant, τ_r is a minimum threshold of radius value, and $N(L_i)$ is the number of pixels labeled with the label L_i on the current image.

For obtained candidate C_p, the score is given by the following formula.

$$Score(C_p) = \frac{N_radius(L_i,C_p)}{N(L_i)}$$

$N_radius(L_i, C_p)$ is the number of pixels labeled L_i in radius $R(L_i)$ from candidate C_p. Therefore, the score is assigned a value between 0 and 1. The higher the score, the higher the probability that the candidate will represent the correct joint position. When the value of $N_radius(L_i, C_p)$ is less than threshold value regardless of $N(L_i)$, the reliability is considered to be too low and the candidate is excluded.

The goal is to select the combination of candidates that maximize the sum of the scores of the selected candidates. Then use them as the final joint positions. If each candidate is independent, then the candidate with the highest score for each label may be selected. There are some constraints that not all candidates are independent and can not be selected together.

First, determine the joint positions of the upper and lower bodies. Because the body part occupies a lot of pixels, it is relatively tolerant to the labeling error. So

that the joint estimation is relatively accurate. Moreover the joint of the body parts is fixed then estimating the joint of the arm and the leg becomes easier.

The remaining joints are divided into four groups as shown below, and the optimal candidate combination is found for each group. (left hand, left radius, left elbow, left humerus, left shoulder, neck) (right hand, right radius, right elbow, right humerus, right shoulder, neck) (left foot, left tibia, left knee, left femur, lower body) (right foot, right tibia, right knee, right femur, lower body) Neck and lowerBody already have been determined in the previous step, and only the combination that includes these joints is considered as valid.

For determining candidate combination of arms and legs, the following three constraints are applied.

distance constraint

It is assumed that the inter-joint distances forming the adjacent rigid bodies in the human model are calculated in advance. Therefore, the joint distance of the hand to elbow, elbow to shoulder, and shoulder to neck is measured in advance. From this distance information, it is possible to calculate the maximum possible distance between two joints that are not immediately adjacent to each other. For example, the maximum distance between hand and shoulder is $D(hand, elbow) + D(elbow, shoulder)$. In this way, the maximum distance between joints can be calculated for joints not adjacent to each other. Using this to find candidate c_p for joint j_p and candidate c_q for joint j_q, if the distance between c_p and c_q exceeds the maximum distance between j_p and j_q.

Line constraint

A line constraint is a constraint between two joints. Adjacent joints are linearly connected to each other on the body(eg. hand to radius, radius to elbow). Therefore, when a virtual line is drawn between two points of adjacent joints in a depth image, there must be some part of the body at that position, and the background depth value should not appear. If candidates c_1 and c_2 of two adjacent joints belong to the same combination, when we draw a line connecting p_1 and p_2 that project c_1 and c_2 to the 2D point on the depth image, You should not have background depth.

Angle constraint

An angle constraint is a constraint among three joints. It is a constraint applied to a joint that forms a rigid body among nonadjacent joints. For example, hand, radius, and elbow must be placed in one rigid body part. Therefore, when candidates C_{hand}, C_{radius}, and C_{elbow} for hand, radius, and elbow are all within a certain range, they can enter the same combination at the same time. In other words, if $\vec{V_1} = C_{hand} -> C_{radius}$, $\vec{V_2} = C_{radius} -> C_{elbow}$, then the following conditions must be met.

$$\vec{V_1} \cdot \vec{V_2} < \tau$$

If C_{hand}, C_{radius}, and C_{elbow} do not satisfy this constraint, all three candidates can not be selected in a combination.

Apply the above three constraints and select the candidate combination with the highest score sum among the combinations satisfying all three constraints for each group. We combine these combinations with previously determined body, neck, and head candidates to determine the final candidate combination. If some joints have no candidates at all, we determined that is a false positive.

Chapter 4

Controllable Sampling Data in the Space of Human Poses

4.1 Overview

Markerless human pose and gesture recognition open up a number of new possibilities in interactive graphics applications and user interface design. The advent of Kinect^{TM}, a motion sensing input device by Microsoft, made it possible to recognize full-body human poses and gestures for practical applications. The hardware of the device includes a depth sensor, which outputs a stream of depth images at a frame rate of 30Hz. The device is equipped with an automatic algorithm that recognizes body parts and labels pixels accordingly in the depth images. The algorithm is based on random decision trees that are learned from a large collection of synthetic body part label images [58]. The synthetic training images are generated using a collection of human motion capture data and assorted human body models.

The effectiveness of the body part recognition algorithm is influenced by the diversity and variability of the motion capture data. For example, if the training data set includes an excessively larger collection of data in one motion category than the other categories, this unbalance would affect the recognition performance negatively. The training data should be collected uniformly from the space of human poses in a carefully planned manner. In practice, this is not easy to accomplish. Motion databases that are available to the public [8, 59] include a lot of locomotion, gestures, and standing actions, which are relatively easy to acquire through marker-based motion capture. However, capturing actions that require the subject to squat, sit, kneel or bend is more challenging due to marker occlusion, and thus such data are not as abundant as standing actions. The recognition algorithm is learned using a large collection of motion data from many motion categories, and sufficient variability may be available only for certain categories.

The recognition algorithm can be learned for specific target applications. The key technology is designing a training data set by mixing motion data with an application-specifically-biased distribution over different categories. For example, a recognition algorithm targeting yoga/stretching should include a wealth of yoga/stretching motions in its training data, which is usually unnecessary for other applications. Yoga/stretching is difficult to capture and thus there is not enough variability in the data set. Simply putting all available data in the training set would make the application-specific data a minority and the algorithm thus trained may not recognize actions in the minority category.

We present a new method that resamples a collection of human motion data to

improve pose variability and achieve an arbitrary level of density in the human pose space. Pose samples should be distributed either uniformly or biased as intended. Human poses are high-dimensional and thus brute-force uniform sampling is intractable. We exploit dimensionality reduction and locally stratified sampling to generate the desired distribution in the human pose space. Our sampling method allows us to manipulate a large data set flexibly to achieve any size, density, and the range of variations.

Our work is largely supplementary to existing pose estimation algorithms. We implemented an algorithm presented by Shotton et al. [58] as our testbed, but our method can be used with other algorithms as well. Our sampling method facilitates the machine learning process to improve the flexibility and versatility of the algorithm. We demonstrate that the algorithm can be learned to recognize challenging poses (for example, sit, kneel, stretching, and yoga positions) by using a remarkably small amount of training data. The algorithm can also be steered to maximize its performance for a specific domain of human poses.

Figure 4.1 shows a overview our work with sampling process. The key component is data resampling at the preprocessing phase. Our resampling algorithm re-distributes training samples uniformly in high-dimensional human pose spaces or in a manner appropriate to specific applications.

4.2 Locally Stratified Sampling

Human poses are high-dimensional, yet highly coordinated. A variety of physical/physiological factors affect how humans pose and determine what poses are nat-

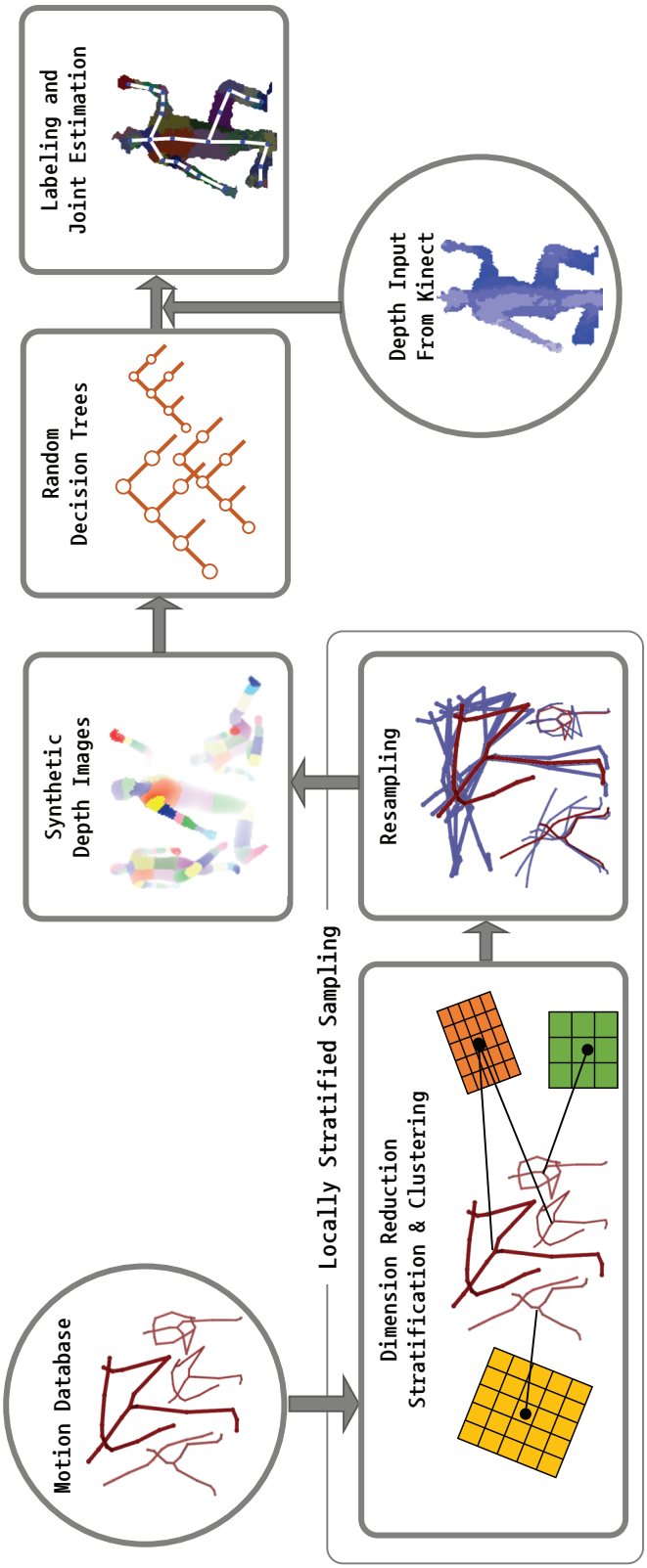


Figure 4.1: *Resampling system overview*

ural and human-like. The pose space is extremely broad, but only a tiny fraction of the space corresponds to natural-looking poses. Natural poses form a low-dimensional sub-manifold in the high-dimensional pose space. Many researchers knows this and accordingly have tried to improve dimension reduction techniques without loss of expression power of human poses in low-dimensional space.

One such strategy is applying local linear coordination on motion data for human tracking [42]. Ideally, we wish to have a collection of training data that cover the space of natural poses comprehensively and uniformly. In practice, the distribution of human pose data collected from public motion databases is domain-specific rather than comprehensive, and severely biased rather than uniform. We need to remove samples from dense regions and add new samples to sparse regions to make the distribution balanced and fill in missing details. Our final goal to locally control the density of human poses with a globally uniform distribution in the space of human poses.

Several techniques for uniform resampling are available. One of the most popular methods is stratified (a.k.a. jittered) sampling [9]. Stratified sampling overlays a grid of cells over the space and takes only one sample at each cell. If more than one sample initially belongs to a cell, one sample is chosen to remain and the others are discarded. If a cell has no samples, a new sample is randomly generated in the cell. Stratified sampling is simple and easy-to-implement, yet effective to reduce clustering of samples.

Applying stratified sampling directly to a collection of human pose data is intractable because of the high-dimensionality of human poses. We address this problem

by exploiting the intrinsic dimension of the pose space and our locally stratified sampling strategy. The key components of our algorithm are dimensionality estimation, stratification of the space, and clustering and resampling of data.

Stratification. A cell in n-dimensional space is a hypercube with an edge length of r. We stratify the space of human poses with a grid of cells. Each cell is supposed to contain at most one pose data in resampling. The size of the cells is related to the density of the output distribution. A smaller r generates a denser distribution of samples. The average distance r_0 from any sample pose to its closest neighbor serves as an initial estimate of the cell size. In our experiments, $r = 2r_0$ unless otherwise specified.

Dimensionality. A large array of literature explores the estimation of intrinsic dimensionality of data [41, 64]. Stratification allows us to estimate intrinsic dimensionality by using a local PCA-based method. The principal component analysis of a data set transforms the data into a new coordinate system spanned by a series of orthogonal vectors, called principal components. The greatest variance of the transformed data lies on the first principal component, the second greatest variance lies on the second principal component, and so on. If the variance on a certain principal component is smaller than the size of a cell, the variance would be discarded in stratified sampling. Therefore, the intrinsic dimensionality of the data set is the number of principal components retaining variance greater than the cell size r.

Clustering. We classify the training data into clusters. Each cluster should have a low intrinsic dimension so that the resampling procedure can be tractable. We use a method based on agglomerative hierarchical clustering: Each sample initially forms

a single cluster, and pairs of clusters are merged incrementally to build a hierarchy of clusters. We prioritize pairs of clusters by max-distance, which is the farthest distance between the members of two clusters. Two clusters of minimal max-distance are first examined for the possibility of merging. The merging is approved if their min-distance, which is the shortest distance between their members, is below a user-specified threshold and their intrinsic dimensionality does not increase beyond the maximum threshold. In our experiments, the threshold for min-distance is $2r$, which makes the samples linked in stratified sampling. The threshold for dimensionality is three.

Resampling. Projecting clusters of data into their low-dimensional PCA space, stratified resampling is performed locally in the PCA space of each individual cluster. Let d be the intrinsic dimension of the cluster. We traverse the cells in a lazy manner (see Figure 4.2): a sample $P_i \in R^d$ in the cluster is selected randomly, and its neighborhood cells are visited for stratification. The neighborhood $N_d(P_i)$ is a d-dimensional grid of cells around P_i. In our experiments, the neighborhood is a grid of either 3^d or 5^d cells. The size of the neighborhood is related to the range and variation of data we would like to achieve through resampling. The cells that have been visited are marked so that we do not need to visit them again. The fill ratio $0 < f \leq 1$ is the ratio of cells holding a sample to the total number of cells. The fill ratio modulates the number of output samples in a continuous scale according to the user's intention.

The pseudocode of our algorithm is provided in Algorithm 1. The algorithm begins by clustering input data into groups (line 1). For each cluster, the samples are projected into a low-dimensional PCA space. We weed out redundant samples from

Algorithm 1 Locally stratified resampling

r : The size of cells

f : The target fill ratio

N_d : A d-dimensional grid of neighborhood cells

$\mathbb{D} = \{P_i\}$: A distribution of input samples (pose vectors)

$\hat{\mathbb{D}}$: A distribution of output samples

$\{\mathbb{C}_j\} \leftarrow$ HierarchicalClustering($\mathbb{D}$);
for each cluster $\mathbb{C}_j$ **do**
  $\hat{\mathbb{C}}_j \leftarrow \emptyset$;
  $d =$ EstimateDimension($\mathbb{C}_j$);
  for each $P_i \in \mathbb{C}_j$ **do**
    for each unmarked cell $\in N_d(P_i)$ **do**
      if the cell is not empty **then**
        Pick a sample $\hat{P}$ randomly in the cell;
        $\hat{\mathbb{C}}_j \leftarrow \hat{\mathbb{C}}_j \cup \{\hat{P}\}$;
      end if
    end for
    while $f <$FillRatio($N_d(P_i)$) **do**
      Pick $\hat{P} \in \hat{\mathbb{C}}_j$ from any unmarked non-empty cell;
      $\hat{\mathbb{C}}_j \leftarrow \hat{\mathbb{C}}_j \setminus \{\hat{P}\}$;
    end while
    while $f >$FillRatio($N_d(P_i)$) **do**
      Pick $\hat{P}$ randomly in any unmarked empty cell;
      if IsValid($\hat{P}$) **then** $\hat{\mathbb{C}}_j \leftarrow \hat{\mathbb{C}}_j \cup \{\hat{P}\}$;
      end if
    end while
    Mark all cells $\in N_d(P_i)$
  end for
end for
$\hat{\mathbb{D}} =$ RemoveCollision($\cup_j \hat{\mathbb{C}}_j$);

crowded cells (lines 6–11). If the fill ratio is above the target number, we randomly remove samples until the ratio drops to the target (lines 12–15). If the fill ratio is below the target, empty cells are randomly chosen to add new samples (lines 16–20). A new sample is a synthesized variant of existing human poses. The synthesized pose may violate joint limits, or may have its body parts interpenetrate with each other or penetrate the ground (line 18). If the penetration depth is below a certain threshold (5cm in our experiments), we use inverse kinematics to push them apart to resolve the interpenetration [39]. If the penetration is deeper, self-collision resolution while maintaining the quality of data is nontrivial. We simply reject such a sample. The last step of the algorithm is to combine samples collected from individual clusters (line 24). The grid of cells of one cluster may overlap with the grid of another cluster in the high-dimensional pose space. We remove collisions so as not to have more than one sample in any cell of any cluster. The bottleneck of the overall procedure is agglomerative hierarchical clustering. The time complexity of clustering is $O(n^2 \log n)$, where n is the number of pose samples. Dimensionality estimation by PCA requires $O(kD^2)$ time, where k is the average size of clusters and D is the average dimensionality of pose data. The time complexity of the whole algorithm is $O(n^2(kD^2 + \log n))$.

4.3 Experimental Results

The motion data generated by our sampling method are used to learn random decision trees, which automatically label input depth images. Each pixel of depth images is labeled according to which body part it belongs to. The final output of the human pose recognition algorithm is reliable proposals for the positions of 3D skeletal

joints. Pixel labeling by a decision tree is quite noisy. We compute joint proposals from a noisy labeled image based on mean shift [58]. We exploit kinematic constraints of the skeleton to improve the robustness and accuracy of joint proposals. The kinematic constraints are derived from the rigidity of bones and their fixed connectivity. For example, a hip joint and a knee joint are connected by a femur, and the distance between the joints is constrained within certain thresholds. The depth values between the connected joints are supposed to measure on the surface of the thigh and therefore should vary linearly from the knee to the hip within an error threshold. These constraints allows us to identify mislabeling of body parts.

The motion capture data are subsampled and classified into three categories. The classification resulted in 8,699 Type I poses, 7,299 Type II poses, and 2,426 Type III poses. The original motion data include a collection of stretching motions of about 13 minutes. Most of the Type III poses come from stretching motions. We used four body models (Male/185cm/70kg, Male/178cm/100kg, Male/179cm/73kg, and Female/158cm/49kg) to generate synthetic depth images at three viewing directions (front, 30 degree left, and 30 degree right), and built three random decision trees for each data set. Technically, exploiting wider variations of human body shapes and viewing directions is not difficult. Learning a decision tree, however is computationally demanding for a large collection of synthetic depth images. Our OpenMP implementation running on 40 cores (Intel Xeon processor E7-4870) can process approximately 10,000 images per hour.

We evaluate the performance of the body part recognition algorithm with respect to pose variability while minimizing the influence of the other conditions. Two mea-

sures, precision $\frac{TP}{TP+FP}$ and accuracy $\frac{TP+TN}{TP+TN+FP+FN}$, are used for the evaluation. Here, a true positive (TP) is an estimated joint located within 10cm from its ground-truth location. A false positive (FP) is an estimated joint located further than 10cm from its ground-truth location. A joint is considered true negative (TN) if the algorithm does not generate its estimated location and the joint is occluded in the depth image. An estimated joint is false negative (FN) if the algorithm fails to locate a visible joint in the depth.

Evaluation using Stanford data. Ganapathi et al. [16] made their data acquired from a time-of-flight camera available on their webpage. The test data come with ground-truth marker locations. The time-of-flight camera has lower resolution (176x144) than a Kinect camera (320x240), and the depth images are noisy and have viewport distortion artifacts. We convert the data to Kinect field-of-view for comparison. Most of the test data are easy to recognize, Type I (upright standing) poses and the data set includes a small number of Type II poses in our classification. Our algorithm was learned from three different sets with fill ratios of 25%, 50%, and 100%. The smallest training set includes 61,000 synthetic images (where the fill ratio is 25%), which is significantly smaller than one million training images of Kinect SDK.

Our algorithm performs comparably to Shotton et al. [58] and outperforms the results of Ganapathi et al. [16] (Figure 4.3). In particular, the test data include high-speed, energetic swings of the arms, for which our algorithm notably outperforms both Shotton's and Ganapathi's algorithms. The precision of upper-body recognition improves with the fill ratio, whereas a higher fill ratio does not result in a higher precision for lower-body recognition. This is because the test poses are mostly upright

standing poses and thus do not have lower-body pose variability that can benefit from a higher density of the training set. On the other hand, training upper-body recognition benefits from the uniformity and higher density of resampled training data to show precision improvement for higher fill ratios.

Evaluation using Type II & III data. We collected our own test data for further comparison with a wider variety of test poses. Our test data consist of 506 real depth images, captured using a Kinect camera and hand-labeled with ground truth joint positions (see Figure 4.4(a) and supplementary material for test images). The test images are collected separately from the motion data used to train the decision trees. We classified the test images into three categories. The classification resulted in 106 test images in Type I, 150 images in Type II, and 250 images in Type III. The comparison using Type II & III data reveals significant improvements of average accuracy over the previous systems. Figure 4.4(b) shows that our algorithm significantly outperforms Kinect SDK for recognizing lower-body joints (31.27%), while the average accuracy for upper-body joints is comparable. To examine the influence of the choice of a threshold value, we plot the mean average accuracy with respect to threshold values in Figure 4.5(a) where both our algorithm and Kinect SDK are applied on our test data. The graph has an inflection point when the threshold value is between 8cm and 10cm. This result is similar to Shotton el al. In our experiments, the threshold is 10cm unless specified otherwise.

Comparison to Other Sampling. In previous work, the size of training data was modulated by subsampling, which removes samples if they are close to their neighboring samples. Two pose samples are considered to be similar if all matching

joints are within a threshold distance. We tested with five threshold values, 2.5cm, 5cm, 7.5cm, 10cm, and 12.5cm, which resulted in 17,279, 12,221, 8,894, 5,567, and 4,086 frames, respectively, after subsampling (Figure 4.5(b)). For fair comparison, we modulated the fill ratio of our algorithm to match the number of samples. In addition, for comparing other sampling method, we applied a Gaussian Mixture Model on the test data. Our resampling method outperforms the subsampling and sampling method regardless of the choice of a threshold value. The plot in the figure shows that our uniform resampling improves the mean average accuracy with more samples, while brute-force subsampling does not. Even the Gaussian Mixture Model shows lower performance as more data is available. This implies that the performance depends on how samples are distributed and the total number of samples is not important. Our uniform resampling makes use of extra samples effectively to achieve performance gain. Figure 4.5(c) shows another comparison between our subsampling and resampling algorithms on Type III testing data. This subsampling is a part of our grid based sampling algorithm (See Algorithm 1 line 6-11). Our algorithm achieves a 10% improvement in accuracy over the original and subsampled training data with a modest increase of the training data. Our algorithm is particularly useful when the training data set is not large enough to model pose variability.

Mixture of Categories. We are particularly interested in understanding how the distribution and mixture ratios of training data affect the performance of body part recognition. To do so, we learned recognition algorithms from training data in each individual pose category (Type I, Type II, and Type III) and mixtures of these categories (Type I&II, Type I&III, Type II&III, and Type I&II&III). The cell size

is $r = 2r_0$. The fill ratios were determined to produce a set of samples of about the same size (approximately 10,000 poses per each data set). These algorithms are applied to Type I, Type II, and Type III test data, respectively, in order to examine the correlation between training and test datasets (see Figure 4.6). As expected, there exists a positive correlation. The algorithm works better for Type-X test data if its training set includes Type-X data. In addition to this basic correlation, the experimental results show both positive and negative synergic effects. The positive synergy means that the algorithm learned from the mixture of Type-X, and Type-Y training data would perform better than the single-type algorithm learned from Type X training data if the Type-Z test data are disjoint from Type-X. The negative synergy indicates an opposite effect. The algorithm learned from the mixture of Type-X, and Type-Y training data would perform worse than the single-type algorithm learned from Type-X training data if the test data are also Type X. In other words, mixing extra data Y would influence the recognition performance positively on average, but negatively for the specific target, assuming that Type-X, Type-Y, and Type-Z are disjoint. The overall performance of a mixed set is better than the overall performance of a single-category set. However, a single category set (for example, a set of Type I training data) outperforms mixed sets (for example, a mixed set of either Type I&II or Type I&III) for the corresponding category of test images, because a mixed set of the same size encodes a wider variety of human poses. Moreover, this tendency also can be seen in superset experiments. They shows slightly better performance because of having more data with respect to combination sets of the same types.

Figure 4.7 show another example of positive synergy. We mixed Type I and Type

III training data with different (Type I : Type III) ratios to build a series of training sets. The overall performance is maximized when the training data are 50% : 50% balanced. Good balance is a key factor to gain better overall performance, even if the domain of training data does not match the domain of test data. The experimental results give us insight as to how to process training data. If we want to design a general purpose algorithm to recognize arbitrary human poses, uniformity across the whole training data would be the most important criterion. If we have a specific application utilizing only a small category of human poses, the training set requires a dense set of relevant pose samples and we have to suppress irrelevant samples to maximize the recognition accuracy.

4.4 Discussion

Our experiments lead to two conclusions. First, the body part recognition algorithm can benefit from uniformly-distributed training data over biased training data if the size of data sets is the same. Second, learning of the body part recognition algorithm can be steered to maximize its recognition performance for a specific category of human poses by providing an appropriate mixture of training data. Large motion databases are cumbersome to handle. Our resampling algorithm provides a convenient means of manipulating a large collection of human pose data. Our algorithm can generate a data set of an arbitrary density, size, and ratio, and an arbitrary range of pose variations.

Our resampling algorithm facilitates the use of many data-driven algorithms. Good examples include style-based inverse kinematics [18], data-driven controller learn-

ing [60], Gaussian process dynamics models [65], and deformable motion models [44]. These methods commonly exploit motion capture data to learn a model of human motion, and therefore can benefit from well-distributed training data.

Currently, the uniformity of motion data has been explored only in the joint angle space. The kinematic skeletal structure is projected onto the image space and then learning is performed with pixel-level image features. Uniformity in the joint angle space may not precisely correspond to uniformity in training images and features. Feature sampling can also be biased; it tends to sample more image features for larger body parts to make recognizing small parts difficult. An interesting direction for future research is to study uniformity in either image spaces or feature spaces. It might be possible to generate uniformly-distributed synthetic depth images and features, which might affect the learning process more immediately.

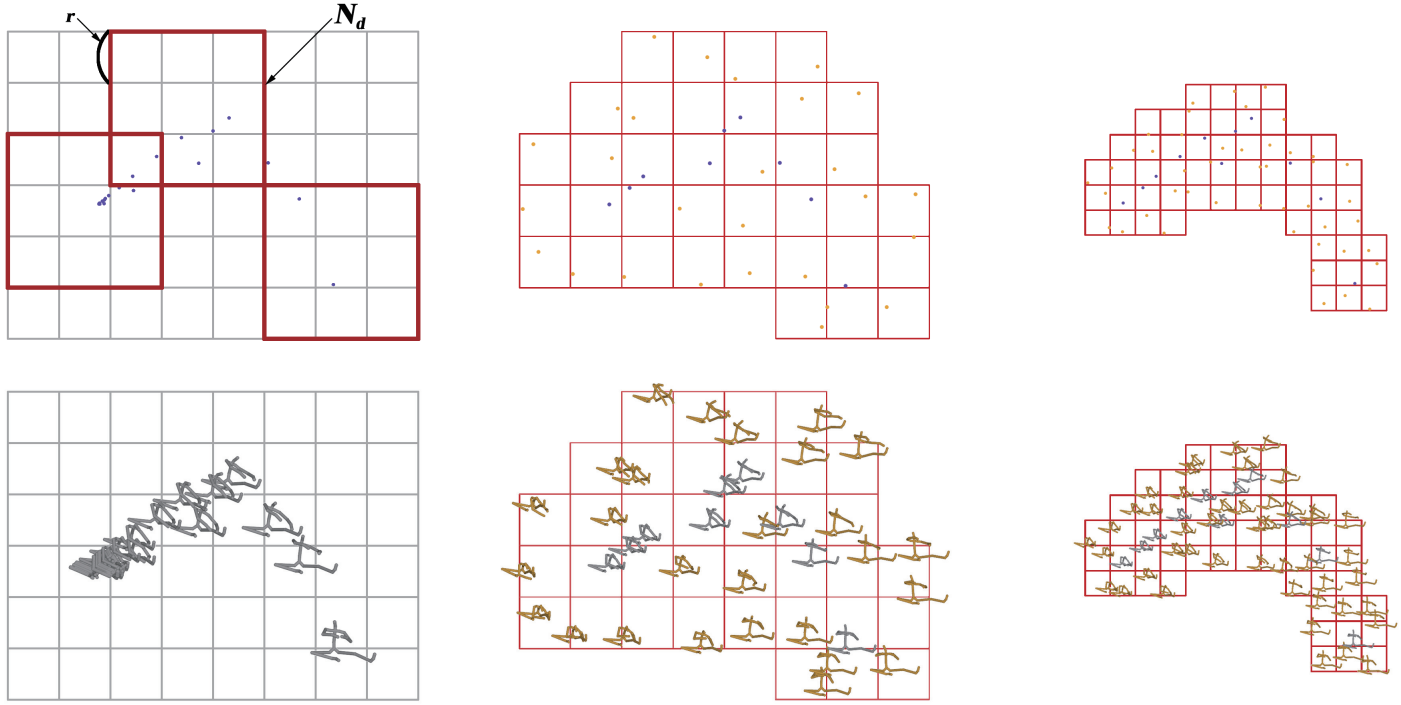


Figure 4.2: *Locally stratified resampling. The pose cluster captured subjects stretching in a sitting position. The top images show full-body poses and the bottom images show pose vectors projected onto a two-dimensional PCA space. The* 3×3 *grid of neighborhood cells are used for local stratification. (Left) Original poses, (Middle) Resampling with a large* r*. The original poses are shown in gray and the new poses synthesized in the resampling process are shown in yellow. (Right) A smaller* r *generates a denser, narrower distribution of output samples.*

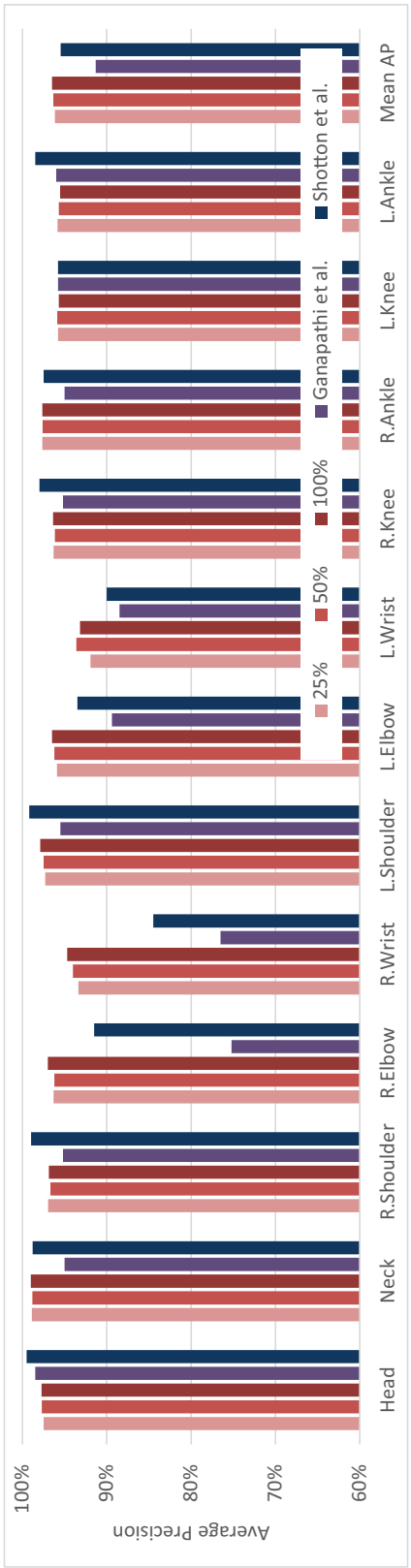
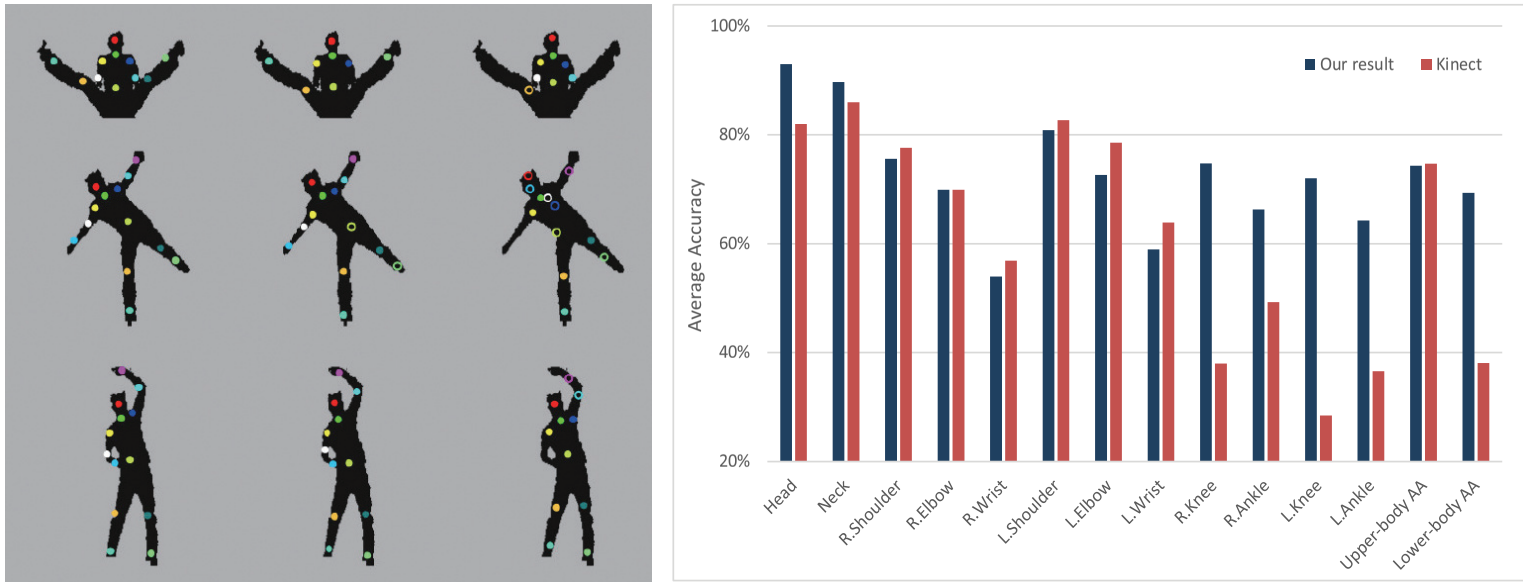
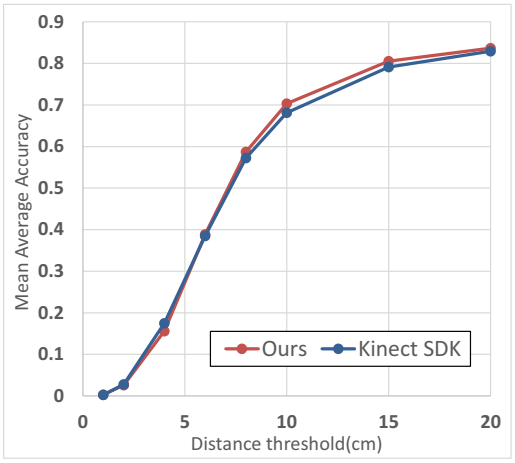


Figure 4.3: *Comparison using Stanford data.*

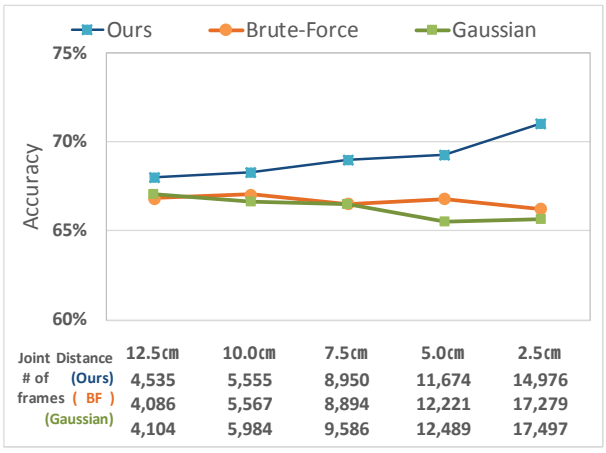


(a) (b)

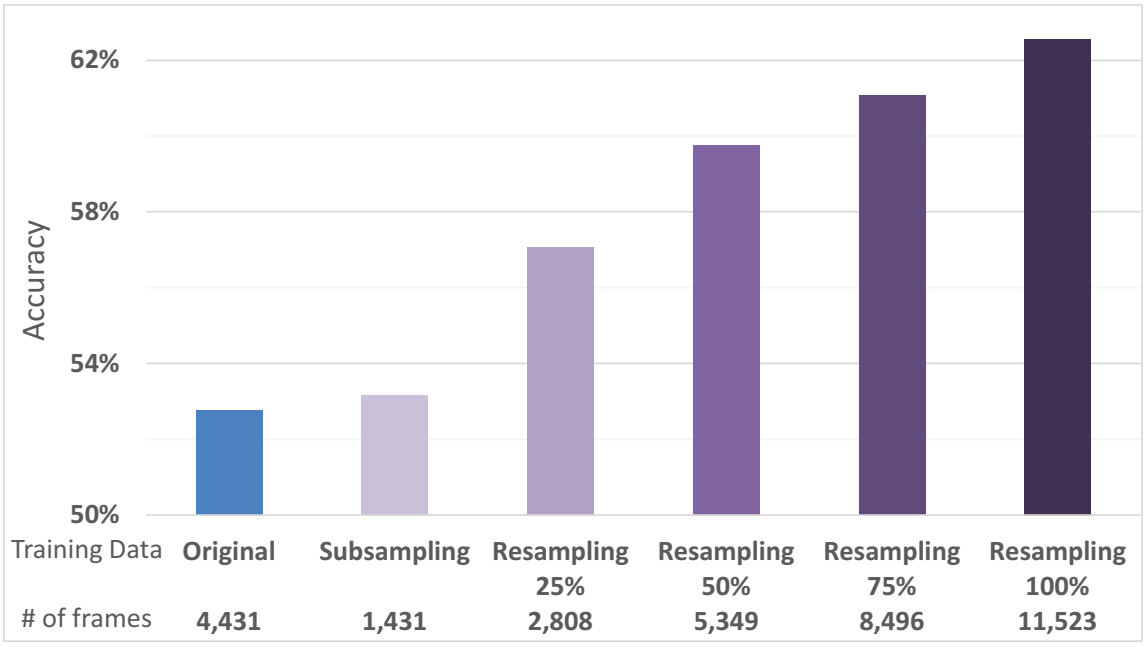
Figure 4.4: *Comparison of results. (a) Comparison of results. (Left) Ground truth, (Middle) Our method, and (Right) Kinect SDK. The solid dots are true positive joint positions and the circles are false positives that are incorrectly labeled. (b) Type II & III data.*



(a)



(b)



(c)

Figure 4.5: *Experimental results. (a) Distance threshold vs. mean average accuracy on our test data. (b) Comparison between brute-force subsampling (S, red plots) and our uniform resampling (R, blue plots). (c) Comparison between our subsampling and resampling.*

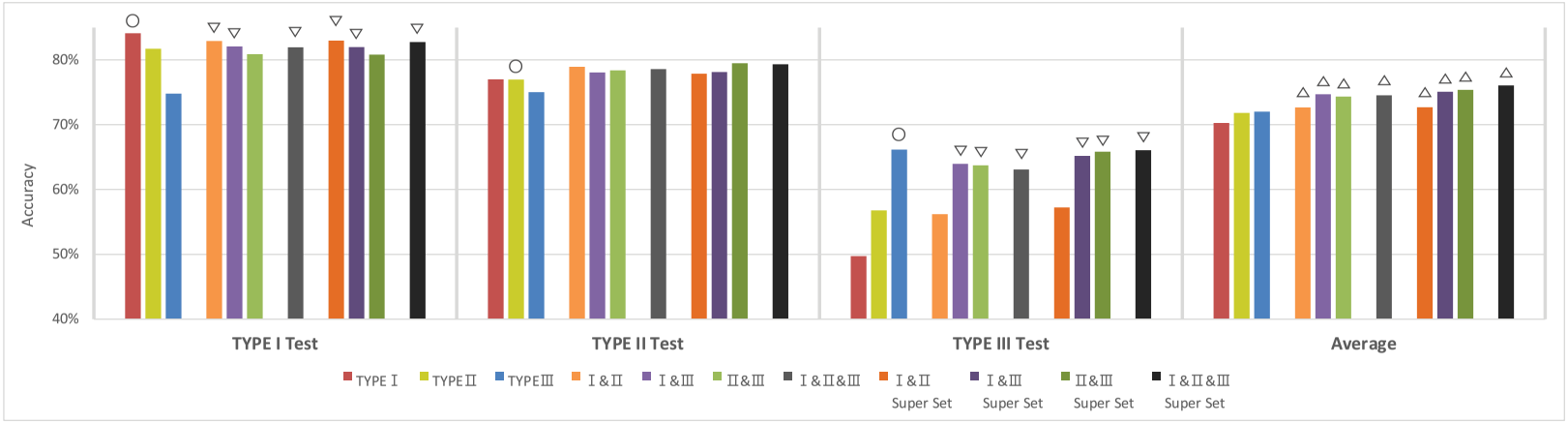


Figure 4.6: *Experimental results: The body part recognition algorithm is learned using each of seven training data sets (Type I, Type II, Type III, and their combination). All data sets are resampled uniformly to have about the same size. The true positive percentage is the ratio of correct joint proposals to the total number of joints. The joint proposal is correct if it is labeled correctly and within 10cm from the ground truth position. The truth positives include no joint proposals if the corresponding joints are not visible in the test image. Symbols* $\circ, \triangle, \bigtriangledown$ *indicate the result demonstrating positive correlation, positive synergy, and negative synergy, respectively. The synergic effects are obvious between Type I and Type III data, which are completely disjoint. Type II data fall in-between Type I and Type III and thus the synergic effects are not apparent.*

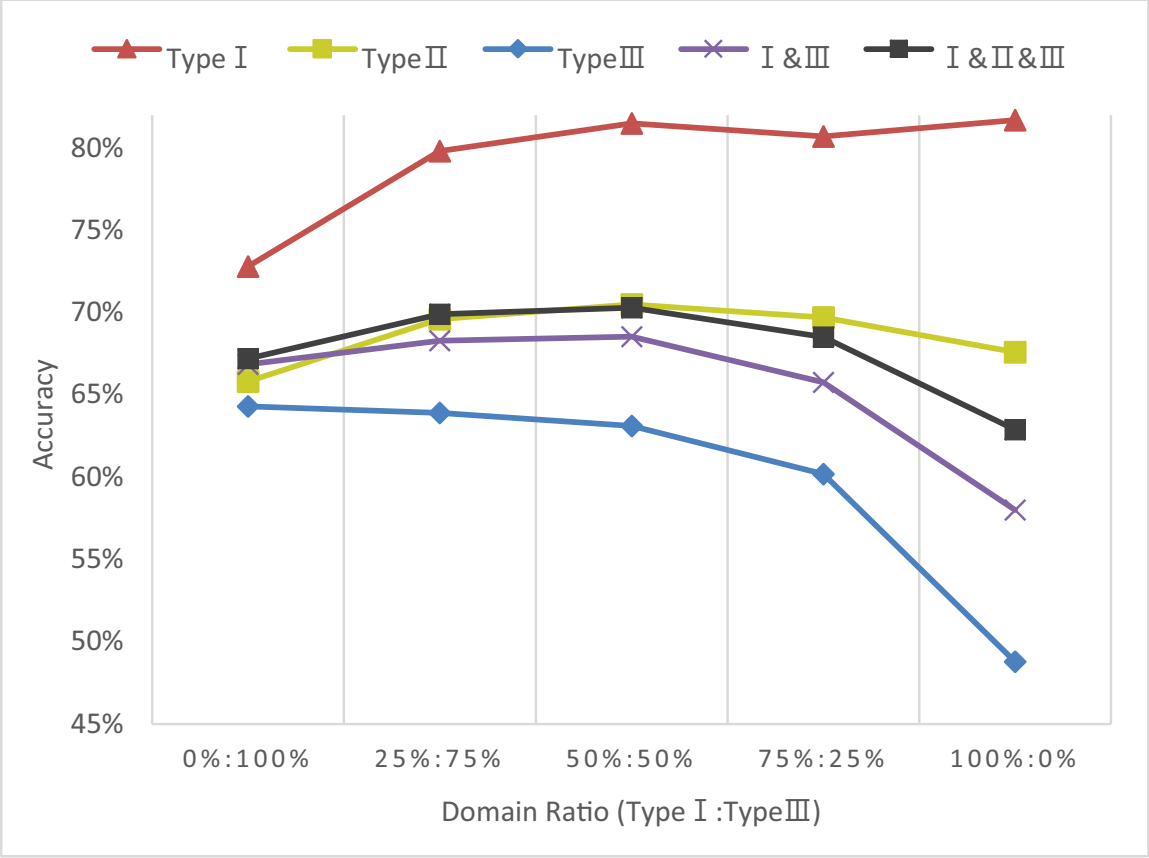


Figure 4.7: *Accuracy plot with respect to the ratio of mixing Type I and Type III training data.*

Chapter 5

Human Pose Estimation with Interacting Prop from Single Depth Image

5.1 Introduction

Recognizing human poses combined with props is important in computer games and virtual environment. Estimation of human body poses from images has been a major goal of computer vision. The availability of depth cameras simplifies and improves the human pose estimation. Shotton et al. [58] developed a body-part recognition algorithm, which is part of the commercial Kinect^{TM} system. This algorithm first classifies different body parts using random decision trees and then estimates the joint positions of human pose from the body-part labels. In Chapter 4, we proposed a sampling method that can design a training set of random decision trees. They generated

various samples of pose data in the space of human poses and controlled the distribution of human poses. Their work made it possible for any user to design the training set, which includes the desired recognition pose data. Girshick et al. [17] estimated the joint positions directly from depth images without body-part labels using an alternative regression-based method. Sun et al. [61] utilized hard constraints of human beings as prior knowledge to improve recognition performance. Wei et al. [67] added full-body tracking to improve the robustness of the algorithm. Sharp et al. [55] improved the hand-tracking accuracy, robustness, and flexibility using generative model fitting.

Since the advent of Kinect™, using the movement of a human body as a controller has been made possible, instead of holding a conventional controller or wearing complex equipment. This technique is a completely new style of controller with a different concept from the previous Joypad. Various games and applications using Kinect™ have been actively developed. Further, Kinect™ has widened its range in various fields such as engineering and medicine by taking advantage of its many possibilities. The possibility of its application is extremely great; therefore, its application area has been expanded to various fields such as game, engineering, and medical areas. For touch-free applications, gesture interface is an attractive point in medical applications because of the sterilization requirements in an operating room [15]. Kinect™ has shown satisfactory results in applications that require a somewhat high level of accuracy. As a result, its applied areas and performance levels have steadily increased

Joypad has been used in conjunction with other accessories in many circumstances. For example, the replica gun-type controller increases immersion and user control

when a user plays a gun-shooting game. Therefore, if the behavior of a user and the props is used to fit a situation, control is expected to be even more spectacular and impressive. If we can utilize an easily available common prop, then usefulness would be further improved. Haggag et al. [20] added prop information into training data; thus, they could label the body parts and prop together. Their work is far from being considered as general props because of the limited view of its performance, the shape of the prop, and the relationship between the prop and a human being.

In this chapter, we present a powerful system that can simultaneously recognize human poses and props using a single depth camera. This system can both estimate human poses and an interacting object and can quickly and precisely express the interaction between the human and objects. This process serves as the basis of a controller system that deals with interactions between humans and environmental objects using depth camera.

5.2 Prop Estimation

In chapter 3, we have discussed the human pose estimation system from a single depth image. This system focuses only on a human pose depth input and is strongly associated with the structure of a human pose. If the depth of the props is added, then the labeling system will fail to label the input depth images. In addition, prop information is not considered. Therefore, to estimate the human pose and interacting props, we need to separate the input depth values into a human pose and other objects. This approach is also considered in the Kinect$^{\text{TM}}$ system [26]. Our approach is described as follows: we assume that D is a set of input depth pixels $\{p_i\}$. If any

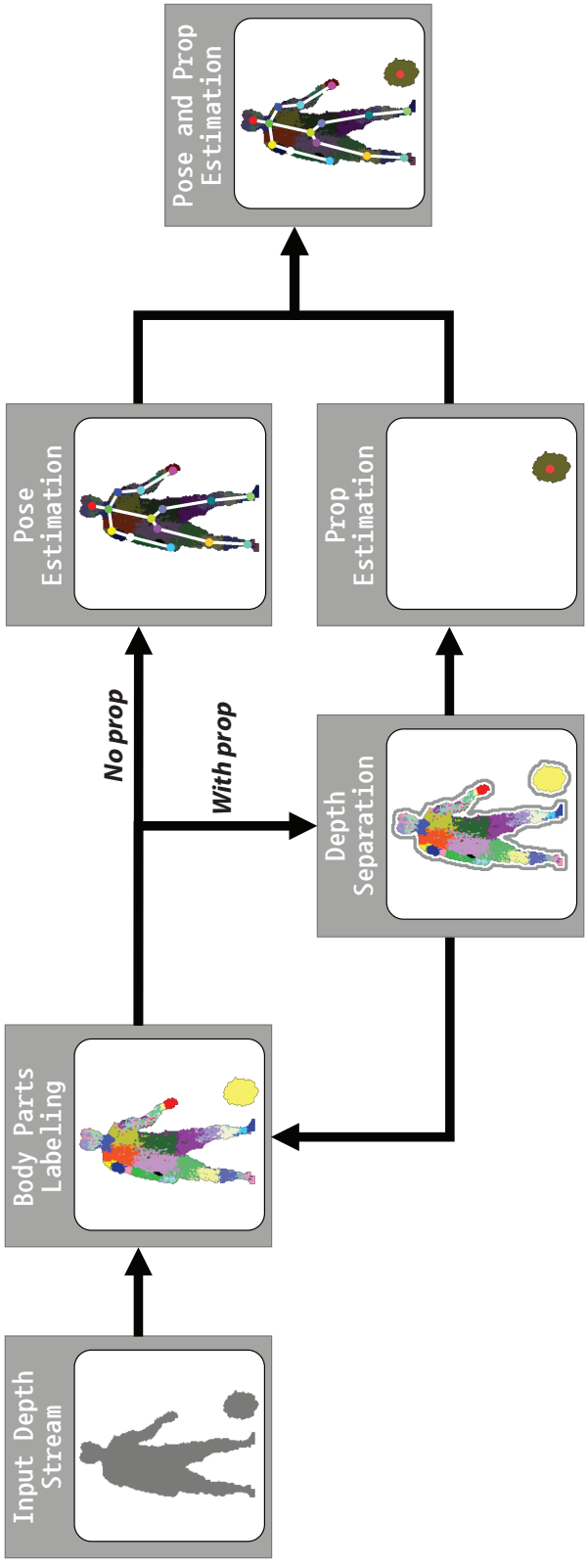


Figure 5.1: *Prop estimation system overview*

other depth input is present except for the human body parts, we divide D into a human pose and other partitions

$$D = D_{pose} \cup D_{others}$$

The depth pixels of the prop belong to the other partition D_{others}. Then, we perform the body-part labeling using D_{pose}.

$$D_{pose \leftarrow label} = LabelEstimator(D_{pose})$$

After labeling the human body parts, we focus on the remaining input depth images. In our experiment, we assume that only a single prop is present. Remaining depth D_{others} is composed of one prop and noises

$$D_{others} = D_{props} \cup D_{noises}$$

Therefore, we apply mean shift clustering to choose one of the most important depth inputs, which would be a prop with a high probability. We apply the joint and prop estimator, which evaluates joint position set jp_i and prop position pp_i, from the labeled depth images.

$$\{jp_i\} = PoseEstimator(D_{pose \leftarrow label})$$

$$\{pp_i\} = PropEstimator(D_{props})$$

If overlaps occur in the human body parts and the prop depth input, it would mean as

$$D_{pose} \cap D_{others} \neq \phi$$

To solve this problem, the failed parts of the joint estimation are considered as props and are substituted for the prop estimation. We differentiate $D_{pose \leftarrow label}$ into the successful and failed parts using the labeling result and model constraints as follows:

$$D_{pose \leftarrow label} = D_{pose \leftarrow success} \cup D_{pose \leftarrow failure}$$

Then, D_{props^*} is defined as follows:

$$D_{props^*} = D_{props} \cup D_{pose \leftarrow failure}$$

We estimate the successful joint position jp_i^* and prop position pp_i^* as

$$\{jp_i^*\} = JointEstimator(D_{pose \leftarrow success})$$

$$\{pp_i^*\} = PropEstimator(D_{props^*})$$

Finally, we combine $\{jp_i^*\}$ and $\{pp_i^*\}$ to figure out the positions of the human pose and prop.

5.3 Experimental Results

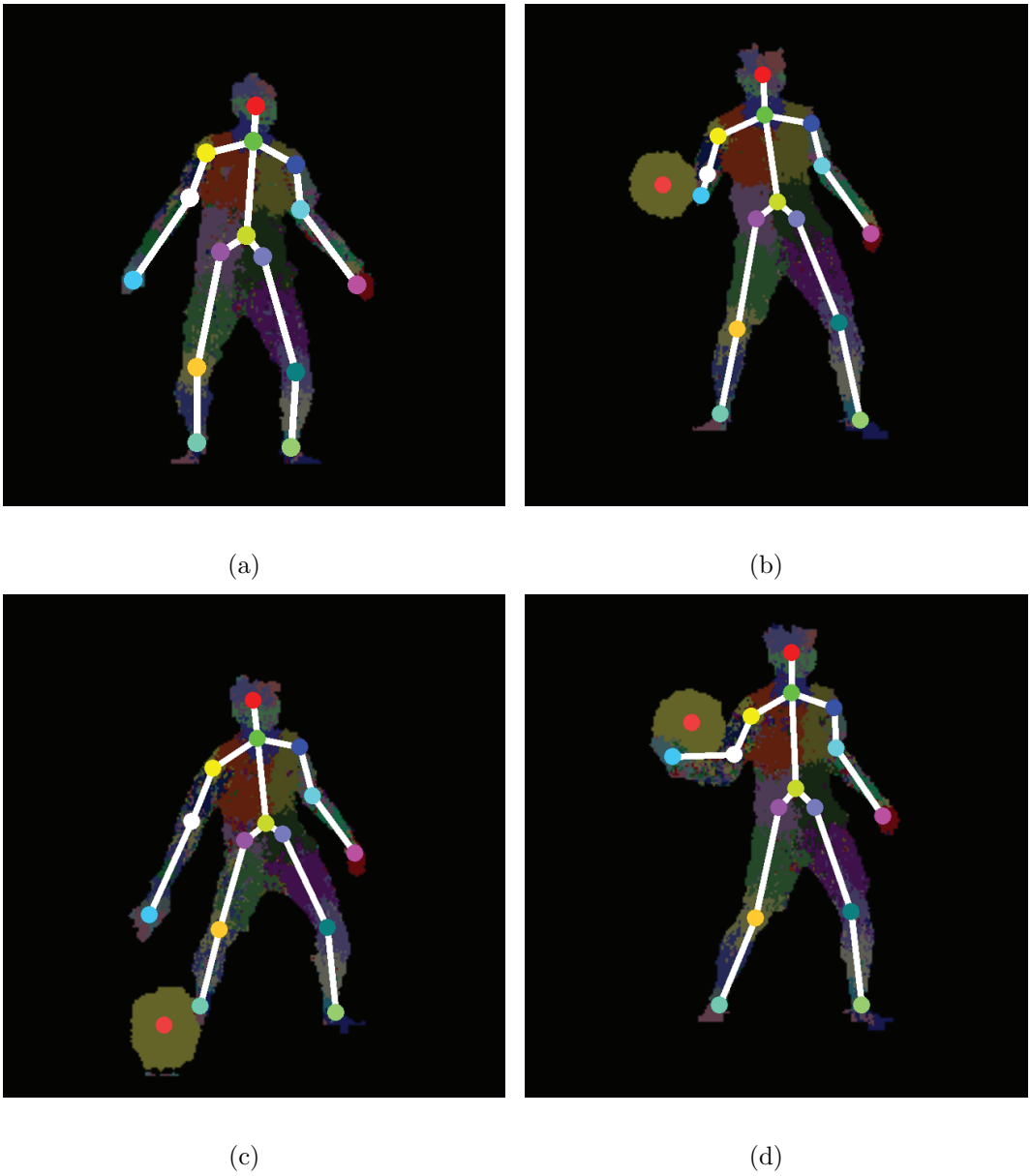
We collected human motion data from public motion databases [8], [59]. To construct the decision trees, we used approximately 10,000 poses. We retargeted the pose data on 185-cm 70-kg human models, which were labeled with 31 body parts. Then, we generated synthetic depth images with body-part label numbers. Each pixel of the depth images was labeled to which body part it belongs to. The synthetic depth images were generated three times more than the pose data because they were captured from three viewing directions (front, 30° and -30°).

The final output from the human pose recognition algorithm was proposed as the positions of the 3D skeletal joints. To realize robustness and accuracy of the joint proposal, we considered the kinematic constraints in estimating the joints. The kinematic constraints were derived from the rigidity of the bones and their fixed connectivity.

We used the Microsoft Kinect^{TM} v1.0 as a depth camera, which has a 320 x 240 depth resolution and 30 frames per second (fps) rate. When real depth input images are incoming, as mentioned in Section 3, pose and prop estimation processes are performed. After applying the pose estimator, we estimated 15 joints for the human poses(See Figure 5.2(a)).

We experimented on a basketball dribbling motion. Figure 4.4(a) shows still images of the input depth stream result, which were obtained using our system. Figure 5.2(a) shows human pose estimation without a prop, and figure 5.2(b)$\sim$ 5.2(d) show human poses with a basketball. Because of the high speed of the bouncing ball with respect to the fps rate of the depth camera, distortion occurred. The results show that our system well performed simultaneous estimation of the human pose and props. In particular, figure 5.2(c) and 5.2(d) show an occurring contacting and slight occluding situation. We also show that an arbitrary object can be estimated(See right side of Figure 5.3).

As shown in results, we experimented the dribbling a basketball motion. Figure 4.4(a) shows still shots of input depth stream result, which were applied with our system. Figure 5.2(a) is a human pose estimation without prop, and figure 5.2(b)$\sim$ 5.2(d) show human pose with a basketball. Due to the high speed of a bounc-



(a) (b)

(c) (d)

Figure 5.2: *Experimental results. (a) Only human pose (b)∼(d) still shots of human pose with bouncing the basketball*

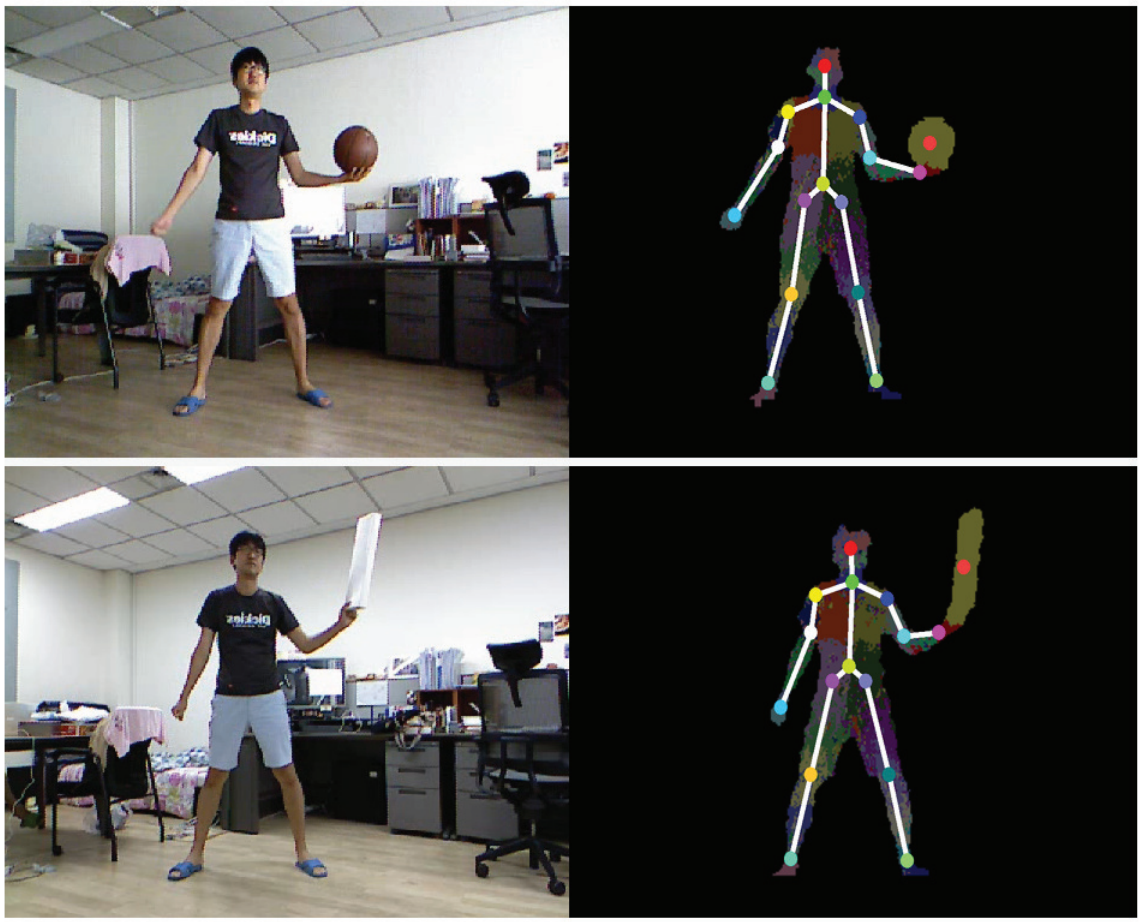


Figure 5.3: *Human pose and props estimation from single depth images (top) holding a basket ball (bottom) holding an arbitrary box*

ing ball with respect to FPS of depth camera, a distortion occurred. As shown as, our system well performed estimation of human pose and props simultaneously. Especially figure 5.2(c) and 5.2(d) show a occurring contacting and slightly occlusion situation. We also show an arbitrary object can be estimated. (See right side of Figure 5.3)

5.4 Discussion

We have proposed a system that can simultaneously recognize human pose and props using a human pose recognition module from a single depth camera. These props are ordinary items, which can be easily obtained from around us. These results show that more props without any special manufacturing process can be used. Our work is widely applicable to various types of controller systems that simultaneously deal with human pose and additional items. Our system has the following limitations; it fails to recognize props if the human body blocks the props. In addition, it fails to recognize when the prop size is smaller than the human body parts, e.g., hand or wrist. One of the solutions to these limitations is to use a sophisticated tracking system. We expect that this technique would more easily help untangle contact problems. In this case, we should use a series of images as an input and not just a single image.

Chapter 6

Enhancing the Estimation of Human Pose from Incomplete Joints

6.1 Overview

It has been an important topic that computer can recognize the human poses at computer vision, computer graphics and many areas. Nowadays, various capture systems enable to recognize human pose. Beginning with optical motion capture system and a variety of methods have been developed, for example inertial system, mechanical system, magnetic system and markerless system.

Among them, using a single depth camera system without markers is spotlighted. This approach received a lot of attention and quickly spread because it is the most convenient way for users to use. A kind of this method has a pre-learning system that

has learned from as many as possible human poses. At runtime, a user pose in front of a single depth camera, real depth image entered, this system estimate human poses using learning system. After whole process, we can generate estimated human joint positions. But, this process is often fail because of mismatch between training data and input data and of property of input poses

We present a new method which can enhance the estimation of human poses. It is very effective and easily apply existing systems. We extract the human pose joint from the depth camera. We trained this joint positions using autoencoder. This system can operate at the existing an invisible or missing joint also. Moreover, this system can work also any other system that uses a joint estimation from human pose estimation system. It can work on real time with pre-learning system, so we can maintain real time pose estimation performance.

6.2 Method

Given a single joint position set with missing joints, we use a autoencoder based method to predict the complete joint positions.

Autoencoder is a kinds of neural network. It usually store compressed feature from input data shape. Human pose data has a extremely high dimensionality and its data lay on complex manifold. The property of autoencoder can generate the result data within manifold of input data. We use this property of autoencoder for applying human pose.

We train autoencoder network with large mount pose data with noise. After training process, we expect this network generate complete and correct joint position from

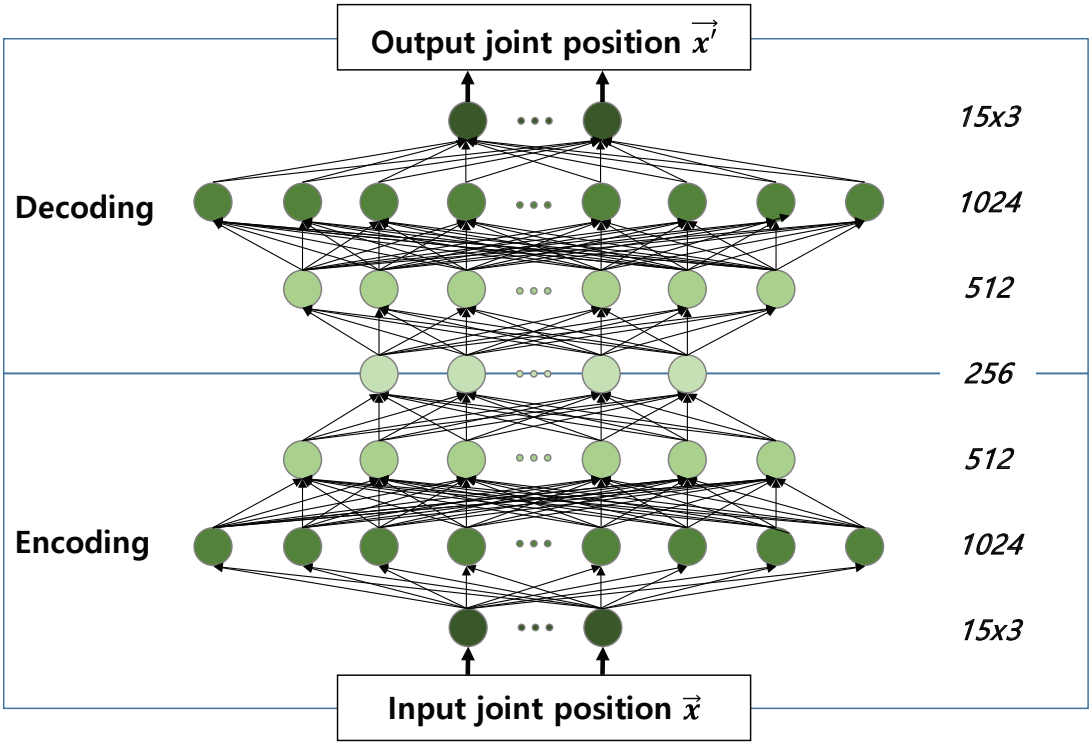


Figure 6.1: *Structure of the autoencoder. Input joint position contains 3x15 values. Layer 2, Layer 3, Layer 4 contains 1024, 512 and 256 nodes.*

incomplete joint position set with missing joint. The range of joint positions value (x_i, y_i, z_i) is like a depth camera input parameter. Therefore a range of each value is depend on performance of depth camera specification. In this work, we use a kinect depth camera, then range is $0 \leq x_i \leq 320$, $0 \leq y_i \leq 240$, $400 \leq z \leq 4000$. We normalize the joint position data. First we matched all pose data align hip position to (0, 0, 0). Then, we normalize position for satisfying scale invariance. Our autoencoder is composed of 3 layers except input layer (See figure 6.1). Let the input joint positions vector $\mathbf{x}$, then whose size expressed $|\mathbf{x}| = 45$ (15 joints and each joint has x, y and z axis values). Each layer is fully connected to adjacent layers, and with increasing

layer order, hidden node size is decreased.

Our input and output data domain lays on nonnegative, than we use a simple matrix multiplication. Let $\mathbf{W}$ is a weight , $\mathbf{b}$ is a bias of each layer, then input of next layer $\Phi(\mathbf{x})$ is following:

$$\Phi(\mathbf{x}) = f(\mathbf{W}\mathbf{x} + \mathbf{b})$$

$$\Phi^{-1}(\mathbf{x}) = f(\mathbf{W}^T\mathbf{X} + \mathbf{b}')$$

Each edge has a softplus function($f(x) = ln(1 + e^x)$) as rectifier function. We use automatic derivatives calculated by TensorFlow [1]. We use a gradient descent method for optimizing reduce the distance cost between output and desired output, and learning rate set to the value of 0.001 [21].

Our autoencoder should be learned joint data with missed joint. There is a study on reconstruction of missing data using auto encoder [45]. In many cases the input data was applied in a continuous signal. On the other hand, we have to deal with missing joint which has a discrete input value. For overcome this problem, we set the input joint position to (0, 0, 0). Then, this input value decay the connected node parameter with setting $\mathbf{W}\mathbf{x}$ is zero. For prevent this situation, we refine the weight of other nodes. Let $\mathbf{W}_{i,j}^{(l)}$ is the parameter associated with the connection between unit j in layer l, and unit i in layer l+1. If p-th joint is missed, then x_p, y_p, z_p are missed. We named the input layer as layer 0, then $\mathbf{W}_{i,3p+k}^{(0)}$ for $0 \leq k \leq 2$ will be affected. As result the value of $\mathbf{W}\mathbf{x}$ set to be zero. We multiply the weight $\frac{|\mathbf{x}|}{|\mathbf{x}|-3}$ by $\mathbf{W}_{i,j}^{(0)}$ for $j \neq 3p + k$ to prevent node parameter from missing joints.

6.3 Experimental Results

We collect human pose data from public motion database [8, 59]. We preprocess human pose with removing the impossible pose(eg. swimming) data in front of camera and with classifying with similar poses(eg. standing pose and sitting pose).

We generate synthetic 3D mesh models which is made by body shape of real human and rigged by pose data. We extract joint position with respect to viewport of depth camera from synthesized models. Finally we collect fifteen joint positions from each poses which are looking forward toward the camera position. We selected the data to use based on the pose data classified in Chapter 3.

Our first autoencoder network is generated from standing poses(Type I). Size of training pose is about 90,000. Second autoencoder network learns from various domain poses for estimate various poses. Type I, II and III poses are used evenly about 10,000 each. These input data are obtained by sampling method described in Chapter 4. Our test pose data is also driven from synthetic 3D mesh models with rigging pose data. Our training process was performed using a GeForce GTX 1080 8GB. Training the network takes about 9 hours when the input data size is about 180,000.

Figure 6.2 shows estimation results using our autoencoder. Left pose is the complete pose. We remove one joint from this ground truth pose(Middle). There is a single missing joint position. In the figure, we can verify that it works properly. Some missing joints are well estimated like ground truth. Figure 6.3 shows failure cases of our network. It can be seen that the predictions on the arms and legs are largely deviated.

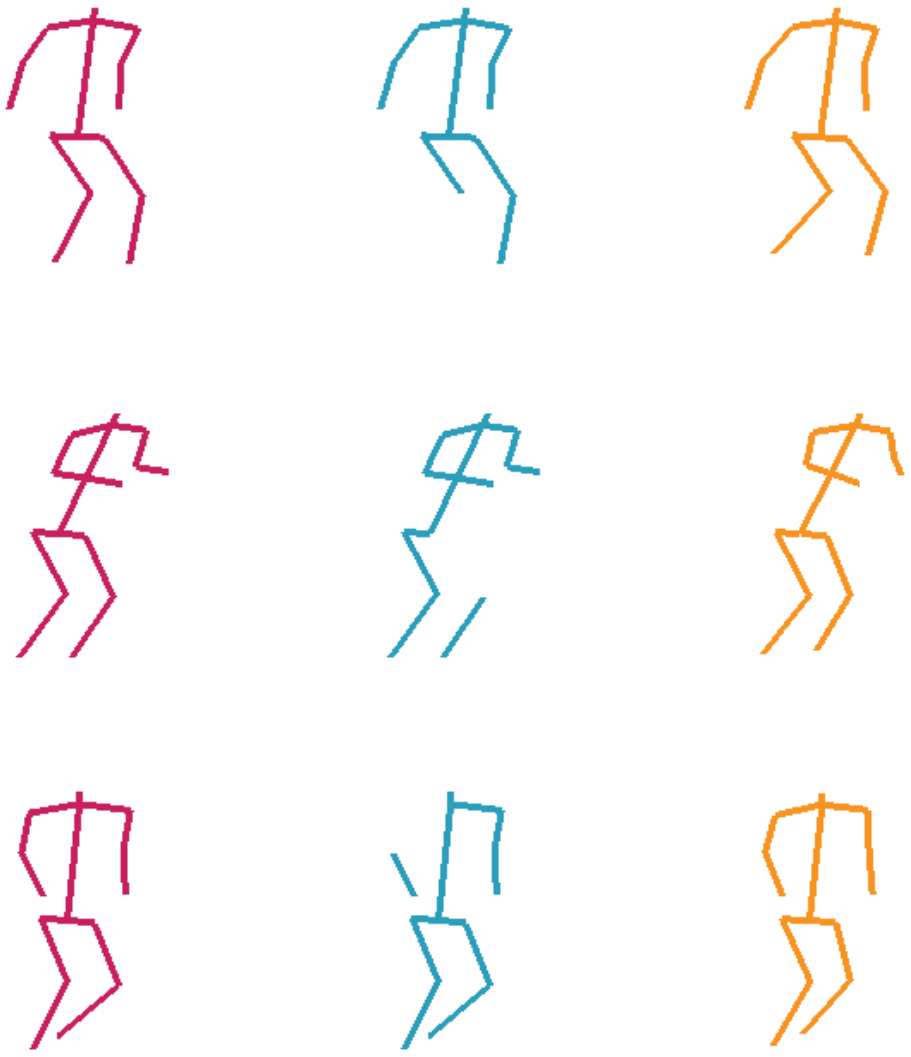


Figure 6.2: *Experimental result. Success case (left) Ground truth (middle) missing input (right) our result.*

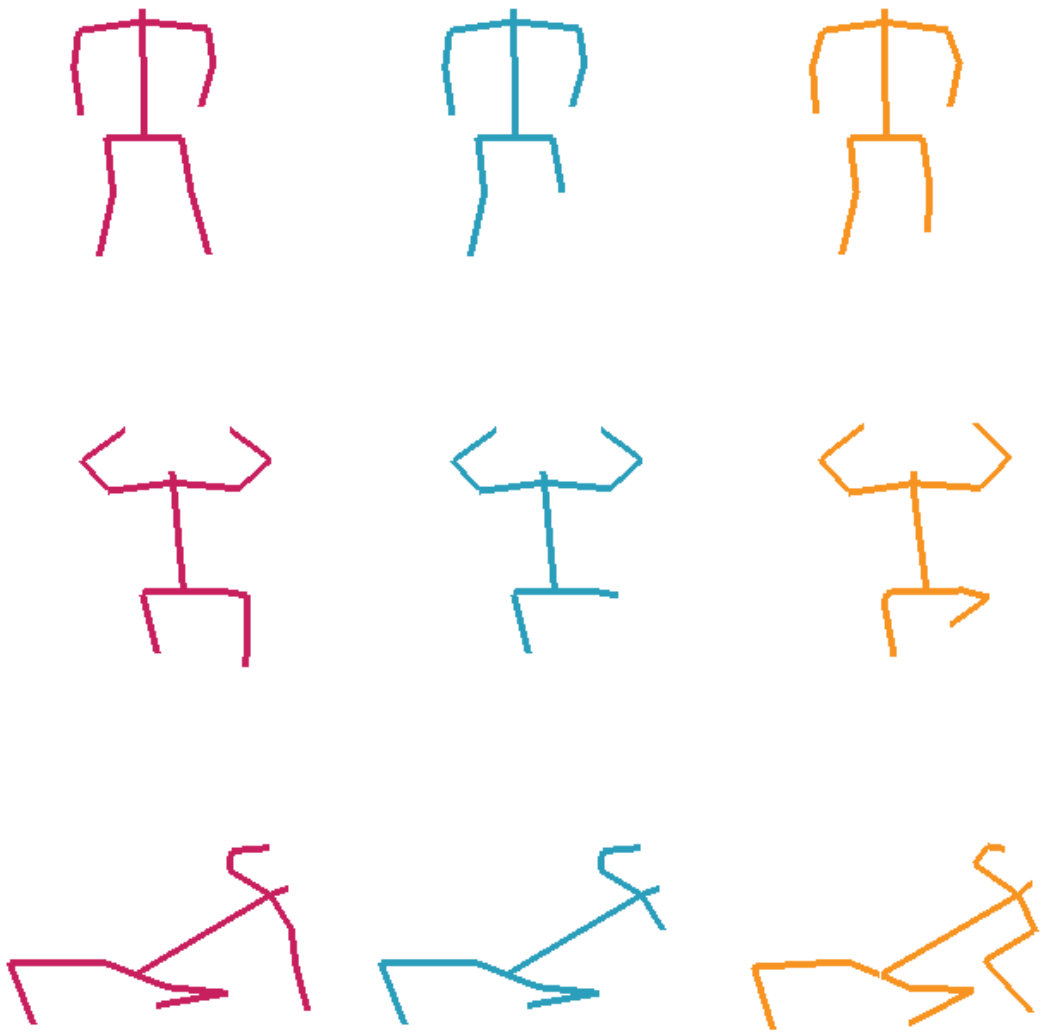


Figure 6.3: *Experimental result. Failure case (left) Ground truth (middle) missing input (right) our result.*

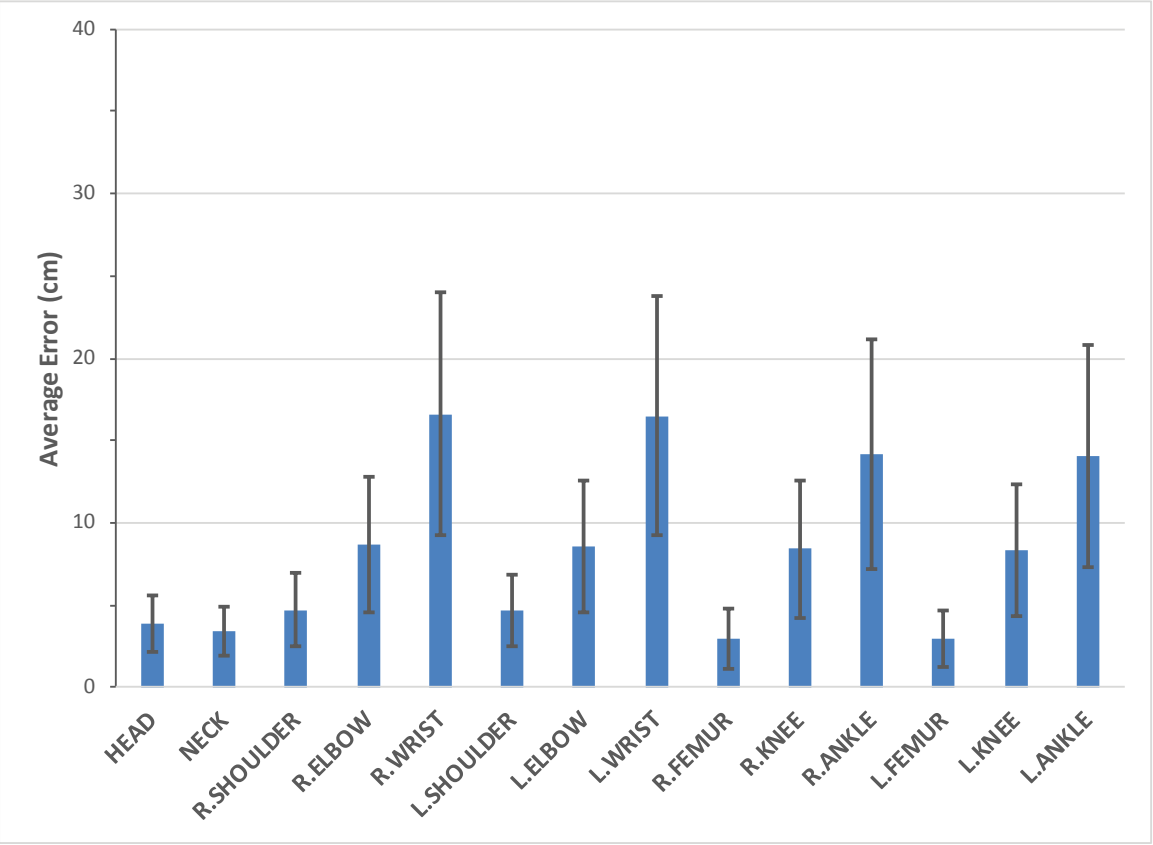


Figure 6.4: *Experimental result. An average error graph of each missing joint*

This result also appear at quantitative results(See figure 6.4). Training set is composed to various domain poses. Each domain has about 10,000 poses for Type I, Type II and Type III and we added their horizontal reflexions. Totally about 60,000 poses are used for autoencoder training. Test set belongs to Type II domain, and its size is 14,647. This error graph shows two meaningful results. First, body line joints (head, neck, shoulder, femur) are well estimated by our network. Their average error is almost bounded within 6.0cm. Second, Like figure 6.3, arms and legs(especially end joints) have a large error value. If the end joint is missed and all the other joints match, it is possible to have multiple positions. Thus missing joint position is extremely overdetermined, estimated joint is far away from the original desired position.

6.4 Discussion

We proposed a method to improve human pose recognition. Our autoencoder network can encode and decode the joint position of human poses. It receives incomplete joint pose data as input and outputs the complete joint pose. This will contribute to improving the performance by estimating the missing joints in the pose estimation system. This system has the advantage of improving performance while maintaining existing posture estimation systems.

In the case of an end joint, there are many choices of joints to be searched. Therefore, it is hard to find exactly correspond desired position with information of the other joint positions. One way to cope with this issue would be to use the depth values around the estimated joints for refining mismatch joint. However, this idea should be applied independently of our network learning.

Chapter 7

Conclusion

In the thesis, we raised the issues about the pose recognition from a single depth images. In many related studies, pose recognition was performed using prior learning data. Therefore it is important to collecting the training data from a huge amount of raw data, but it is hard work because of data size and dimension size of data. Many human recognition system from single depth images always suffer from noise data and invisible body parts. Therefore, it should be made up for in various way.

To cope with these problems, we addressed three ways to get over issues that can occur while performing pose recognition. First we can control and generate human poses in the space of human poses. Nevertheless human pose data has extremely high dimension, we define pose data group at the low dimension space which can express human poses well. It made unbalanced pose data to balanced pose data with desired pose density. It can helps all the other pose database system construction with satisfying desired pose distribution. Second, our system can recognize arbitrary props with interacting human. It has a advantage that perform no annotation or pre-

labeling. Last, we shows a enhanced pose estimation system that using neural network system. Our autoencoder network has trained for recovering complete human joint pose from incomplete or unstable input pose data. This system can make it possible that increase the performance of any other estimation system.

We showed a possibility to enable sampling high-dimensional pose data in the space of human poses. To other field can apply our sampling method, especially to the character animation. For the more worthy apply, we should extend our sampling algorithm operates in the space of human poses to operate in the space of human poses and continuous motion in the future. It will not only generate good quality data for performing other data-driven learning data-based works, but we can utilize a result of motion sampling directly. It is very useful because people usually need a human motion rather than a single pose, for example retrieval motion, editing motion, and synthesis motions. It is very useful because people often need a human motion data rather than a single pose data.

Through the thesis, we have made great efforts to computer understand the human poses. We would make it possible to computer understand human motions, furthermore understand human behavior and enable to behavior capture.

Bibliography

[1] Martın Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous systems, 2015. *Software available from tensorflow. org*, 1, 2015.

[2] Ossama Abdel-Hamid, Abdel-Rahman Mohamed, Hui Jiang, Li Deng, Gerald Penn, and Dong Yu. Convolutional neural networks for speech recognition. *IEEE/ACM Trans. Audio, Speech and Lang. Proc.*, 22(10):1533–1545, October 2014.

[3] Okan Arikan and D. A. Forsyth. Interactive motion generation from examples. *ACM Trans. Graph.*, 21(3):483–490, July 2002.

[4] Andreas Baak, Meinard Muller, Gaurav Bharaj, Hans-Peter Seidel, and Christian Theobalt. A data-driven approach for real-time full body pose reconstruction from a depth camera. In *Proceedings of the 2011 International Conference on Computer Vision*, pages 1092–1099, 2011.

[5] Federica Bogo, Angjoo Kanazawa, Christoph Lassner, Peter V. Gehler, Javier

Romero, and Michael J. Black. Keep it SMPL: automatic estimation of 3d human pose and shape from a single image. *CoRR*, abs/1607.08128, 2016.

[6] Thomas Brox, Bodo Rosenhahn, Daniel Cremers, and Hans-Peter Seidel. Nonparametric density estimation with adaptive, anisotropic kernels for human motion tracking. In *Proceedings of the 2nd conference on Human motion: understanding, modeling, capture and animation*, pages 152–165, 2007.

[7] Yizong Cheng. Mean shift, mode seeking, and clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):790–799, Aug 1995.

[8] CMU-DB. Carnegie Mellon University motion database. http://mocap.cs.cmu.edu/.

[9] Robert L. Cook. Stochastic sampling in computer graphics. *ACM Transactions on Graphics*, 5(1):51–72, 1986.

[10] Mingsong Dou, Sameh Khamis, Yury Degtyarev, Philip Davidson, Sean Fanello, Adarsh Kowdle, Sergio Orts Escolano, Christoph Rhemann, David Kim, Jonathan Taylor, Pushmeet Kohli, Vladimir Tankovich, and Shahram Izadi. Fusion4d: Real-time performance capture of challenging scenes. In *ACM SIGGRAPH Conference on Computer Graphics and Interactive Techniques*, 2016.

[11] Yu Du, Yongkang Wong, Yonghao Liu, Feilin Han, Yilin Gui, Zhen Wang, Mohan Kankanhalli, and Weidong Geng. *Marker-Less 3D Human Motion Capture with Monocular Image Sequence and Height-Maps*, pages 20–36. Springer International Publishing, Cham, 2016.

[12] Daniel Dunbar and Greg Humphreys. A spatial data structure for fast poisson-disk sample generation. *ACM Transactions on Graphics (SIGGRAPH 2006)*, 25(3):503–508, 2006.

[13] Jialue Fan, Wei Xu, Ying Wu, and Yihong Gong. Human tracking using convolutional neural networks. *Trans. Neur. Netw.*, 21(10):1610–1623, October 2010.

[14] K. Forbes and E. Fiume. An efficient search algorithm for motion data using weighted pca. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '05, pages 67–76, New York, NY, USA, 2005. ACM.

[15] L. Gallo, A. P. Placitelli, and M. Ciampi. Controller-free exploration of medical image data: Experiencing the kinect. In *Proceedings of the 2011 24th International Symposium on Computer-Based Medical Systems (CBMS)*, pages 1–6, 2011.

[16] Varun Ganapathi, Christian Plagemann, Sebastian Thrun, and Daphne Koller. Real time motion capture using a single time-of-flight camera. In *CVPR*, 2010.

[17] Ross Girshick, Jamie Shotton, Pushmeet Kohli, Antonio Criminisi, and Andrew Fitzgibbon. Efficient regression of general-activity human poses from depth images. In *Proceedings of the 2011 International Conference on Computer Vision (ICCV)*, pages 415–422, 2011.

[18] Keith Grochow, Steven L. Martin, Aaron Hertzmann, and Zoran Popović. Style-

based inverse kinematics. *ACM Transactions on Graphics (SIGGRAPH 2004)*, 23(3):522–531, 2004.

[19] Sehoon Ha, Yunfei Bai, and C. Karen Liu. Human motion reconstruction from force sensors. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '11, pages 129–138, New York, NY, USA, 2011. ACM.

[20] H. Haggag, M. Hossny, S. Nahavandi, S. Haggag, and D. Creighton. Body parts segmentation with attached props using rgb-d imaging. In *Digital Image Computing: Techniques and Applications (DICTA), 2015 International Conference on*, pages 1–8, Nov 2015.

[21] Geoffrey E. Hinton. *A Practical Guide to Training Restricted Boltzmann Machines*, pages 599–619. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[22] Daniel Holden, Jun Saito, and Taku Komura. A deep learning framework for character motion synthesis and editing. *ACM Trans. Graph.*, 35(4):138:1–138:11, July 2016.

[23] Daniel Holden, Jun Saito, Taku Komura, and Thomas Joyce. Learning motion manifolds with convolutional autoencoders. In *SIGGRAPH Asia 2015 Technical Briefs*, SA '15, pages 18:1–18:4, New York, NY, USA, 2015. ACM.

[24] C. Hong, J. Yu, J. Wan, D. Tao, and M. Wang. Multimodal deep autoencoder for human pose recovery. *IEEE Transactions on Image Processing*, 24(12):5659–5670, Dec 2015.

[25] Autodesk Inc. Motionbuilder.

[26] S. Izadi, J. Shotton, J. Winn, A. Criminisi, O. Hilliges, M. Cook, and D. Molyneaux. Detection of body and props, February 25 2014. US Patent 8,660,303.

[27] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, and Andrew Fitzgibbon. Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th annual ACM symposium on User interface software and technology (UIST)*, pages 559–568, 2011.

[28] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3d convolutional neural networks for human action recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(1):221–231, January 2013.

[29] Ho Yub Jung, Soochahn Lee, Yong Seok Heo, and Il Dong Yun. Random tree walk toward instantaneous 3d human pose estimation. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2467–2474, June 2015.

[30] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '14, pages 1725–1732, Washington, DC, USA, 2014. IEEE Computer Society.

[31] Cem Keskin, Furkan Kıraç, Yunus Emre Kara, and Lale Akarun. Hand pose estimation and hand shape classification using multi-layered randomized decision forests. In *Proceedings of the 12th European conference on Computer Vision - Volume Part VI*, ECCV'12, pages 852–863, 2012.

[32] Jongmin Kim, Yeongho Seol, and Jehee Lee. Human motion reconstruction from sparse 3d motion sensors using kernel cca-based regression. *Comput. Animat. Virtual Worlds*, 24(6):565–576, November 2013.

[33] Lucas Kovar and Michael Gleicher. Automated extraction and parameterization of motions in large data sets. *ACM Trans. Graph.*, 23(3):559–568, August 2004.

[34] Lucas Kovar, Michael Gleicher, and Frédéric Pighin. Motion graphs. *ACM Trans. Graph.*, 21(3):473–482, July 2002.

[35] Taesoo Kwon and Sung Yong Shin. Motion modeling for on-line locomotion synthesis. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '05, pages 29–38, New York, NY, USA, 2005. ACM.

[36] Manfred Lau, Ziv Bar-Joseph, and James Kuffner. Modeling spatial and temporal variation in motion data. *ACM Transactions on Graphics (SIGGRAPH Asia 2009)*, 28(5), 2009.

[37] Neil Lawrence and Aapo Hyvärinen. Probabilistic non-linear principal component analysis with gaussian process latent variable models. *Journal of Machine Learning Research*, 6:1783–1816, 2005.

[38] Jehee Lee, Jinxiang Chai, Paul S. A. Reitsma, Jessica K. Hodgins, and Nancy S. Pollard. Interactive control of avatars animated with human motion data. *ACM Trans. Graph.*, 21(3):491–500, July 2002.

[39] Jehee Lee and Sung Yong Shin. A hierarchical approach to interactive motion editing for human-like figures. In *Proceedings of SIGGRAPH 99*, pages 39–48, 1999.

[40] Kang Hoon Lee, Myung Geol Choi, and Jehee Lee. Motion patches: building blocks for virtual environments annotated with motion data. *ACM Transactions on Graphics (SIGGRAPH 2006)*, 26(3), 2006.

[41] E. Levina and P.J. Bickel. Maximum likelihood estimation of intrinsic dimension. In *Advances in Neural Information Processing Systems (NIPS) 17*, 2005.

[42] Rui Li, Ming-Hsuan Yang, Stan Sclaroff, and Tai-Peng Tian. Monocular tracking of 3d human motion with a coordinated mixture of factor analyzers. In *ECCV (2)*, pages 137–150, 2006.

[43] Z. Liu, L. Zhou, H. Leung, and H. P. H. Shum. Kinect posture reconstruction based on a local mixture of gaussian process models. *IEEE Transactions on Visualization and Computer Graphics*, 22(11):2437–2450, Nov 2016.

[44] Jianyuan Min, Yen-Lin Chen, and Jinxiang Chai. Interactive generation of human animation with deformable motion models. *ACM Transactions on Graphics*, 29(1), 2009.

[45] V. Miranda, J. Krstulovic, H. Keko, C. Moreira, and J. Pereira. Reconstructing

missing data in state estimation with autoencoders. *IEEE Transactions on Power Systems*, 27(2):604–611, May 2012.

[46] Fabian Nasse, Christian Thurau, and Gernot A. Fink. Face detection using gpu-based convolutional neural networks. In *Proceedings of the 13th International Conference on Computer Analysis of Images and Patterns*, CAIP '09, pages 83–90, Berlin, Heidelberg, 2009. Springer-Verlag.

[47] Nam Nguyen, Nkenge Wheatland, David Brown, Brian Parise, C. Karen Liu, and Victor Zordan. Performance capture with physical interaction. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '10, pages 189–195, Aire-la-Ville, Switzerland, Switzerland, 2010. Eurographics Association.

[48] Iason Oikonomidis, Nikolaos Kyriazis, and Antonis Argyros. Efficient model-based 3d tracking of hand articulations using kinect. In *Proceedings of the British Machine Vision Conference*, pages 101.1–101.11, 2011.

[49] Sang Il Park, Hyun Joon Shin, and Sung Yong Shin. On-line locomotion generation based on motion blending. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '02, pages 105–111, New York, NY, USA, 2002. ACM.

[50] Katherine Pullen and Christoph Bregler. Motion capture assisted animation: Texturing and synthesis. *ACM Trans. Graph.*, 21(3):501–508, July 2002.

[51] Zhou Ren, Jingjing Meng, Junsong Yuan, and Zhengyou Zhang. Robust hand

gesture recognition with kinect sensor. In *Proceedings of the 19th ACM international conference on Multimedia*, pages 759–760, 2011.

[52] Helge Rhodin, Christian Richardt, Dan Casas, Eldar Insafutdinov, Mohammad Shafiei, Hans-Peter Seidel, Bernt Schiele, and Christian Theobalt. Egocap: Egocentric marker-less motion capture with two fisheye cameras. *ACM Trans. Graph.*, 35(6):162:1–162:11, November 2016.

[53] Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, 2000.

[54] Shahram Izadi Pushmeet Kohli David Kim David Sweeney Antonio Criminisi Jamie Shotton Sing Bing Kang Tim Paek Sean Fanello, Cem Keskin. Learning to be a depth camera for close-range human capture and interaction. *ACM Transactions on Graphics (TOG) - Proceedings of ACM SIGGRAPH 2014*, 33, July 2014.

[55] Toby Sharp, Cem Keskin, Duncan Robertson, Jonathan Taylor, Jamie Shotton, David Kim, Christoph Rhemann, Ido Leichter, Alon Vinnikov, Yichen Wei, Daniel Freedman, Pushmeet Kohli, Eyal Krupka, Andrew Fitzgibbon, and Shahram Izadi. Accurate, robust, and flexible real-time hand tracking. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pages 3633–3642, New York, NY, USA, 2015. ACM.

[56] Hyun Joon Shin and Jehee Lee. Motion synthesis and editing in low-dimensional spaces. *Computer Animation and Virtual Worlds*, 17(3-4):219–227, 2006.

[57] Hyun Joon Shin and Jehee Lee. Motion synthesis and editing in low-dimensional spaces. *Computer Animation and Virtual Worlds*, 17(3-4):219–227, July 2006.

[58] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth images. In *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1297–1304, 2011.

[59] SNU-DB. Seoul National University motion database. http://mrl.snu.ac.kr/ mdb/.

[60] Kwang Won Sok, Manmyung Kim, and Jehee Lee. Simulating biped behaviors from human motion data. *ACM Transactions on Graphics (SIGGRAPH 2007)*, 26(3), 2007.

[61] Min Sun, Pushmeet Kohli, and Jamie Shotton. Conditional regression forests for human pose estimation. In *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3394–3401, 2012.

[62] Graham W. Taylor, Geoffrey E. Hinton, and Sam T. Roweis. Two distributed-state models for generating high-dimensional time series. *J. Mach. Learn. Res.*, 12:1025–1068, July 2011.

[63] Jonathan Tompson, Murphy Stein, Yann Lecun, and Ken Perlin. Real-time continuous pose recovery of human hands using convolutional networks. *ACM Trans. Graph.*, 33(5):169:1–169:10, September 2014.

[64] P.J. Verveer and R.P.W. Duin. An evaluation of intrinsic dimensionality estimators. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(1):81 –86, 1995.

[65] Jack M. Wang, David J. Fleet, and Aaron Hertzmann. Gaussian process dynamical models for human motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):283–298, 2008.

[66] Yingying Wang and Michael Neff. Deep signatures for indexing and retrieval in large motion databases. In *Proceedings of the 8th ACM SIGGRAPH Conference on Motion in Games*, MIG '15, pages 37–45, New York, NY, USA, 2015. ACM.

[67] Xiaolin Wei, Peizhao Zhang, and Jinxiang Chai. Accurate realtime full-body motion capture using a single depth camera. *ACM Transactions on Graphics (SIGGRAPH Asia 2012)*, 31(6), 2012.

[68] Thibaut Weise, Sofien Bouaziz, Hao Li, and Mark Pauly. Realtime performance-based facial animation. *ACM Transactions on Graphics (SIGGRAPH 2011)*, 30(4), 2011.

[69] Makoto Yamada, Leonid Sigal, and Michalis Raptis. No bias left behind: Covariate shift adaptation for discriminative 3d pose estimation. In *ECCV (4)*, pages 674–687, 2012.

[70] Mao Ye, Xianwang Wang, Ruigang Yang, Liu Ren, and M. Pollefeys. Accurate 3d pose estimation from a single depth image. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 731–738, 2011.

[71] Peizhao Zhang, Kristin Siu, Jianjie Zhang, C. Karen Liu, and Jinxiang Chai. Leveraging depth cameras and wearable pressure sensors for full-body kinematics and dynamics capture. *ACM Trans. Graph.*, 33(6):221:1–221:14, November 2014.

초 록

하나의 깊이 카메라를 이용하여 마커를 부착하지 않은 사람의 자세를 인식하는 문제(markerless pose recognition)는 대화식 그래픽 응용 프로그램 및 사용자 인터페이스 디자인에서 중요한 역할을 한다. 최근의 자세 인식 알고리즘은 거대한 양의 모션 캡쳐 데이터를 활용하는 기계 학습 방법을 많이 이용한다. 이러한 알고리즘의 효과는 학습 데이터의 다양성과 변동성에 크게 영향을 받는다. 이러한 자세 인식 시스템을 활용하기 위해 사람의 자세를 컨트롤러로 사용하도록 하는 많은 어플리케이션이 개발되었다. 많은 경우, 주변에 흔히 있는 일반적인 소품을 함께 사용하면 몰입하여 제어를 수행하는 데 도움이 된다. 그럼에도 불구하고, 아직 사람 자세와 소품 인식을 함께 인식하는 시스템은 아직 성공적으로 수행되지 못하고 있다. 또한, 하나의 깊이 카메라를 사용하면 카메라에 보이지 않는 부분은 관측된 데이터가 없어 사람의 자세 인식 품질을 낮추는 등의 문제들이 있다.

본 논문에서는 하나의 깊이 카메라에서 성공적으로 사람의 자세를 추정할 수 있도록 인간의 동작 데이터를 다루는 방법을 제시한다. 먼저 우리는 사람의 자세 데이터를 재생성하여 자세 변동성을 개선하고, 사람 자세의 공간에서 임의의 크기와 밀도 수준을 달성하는 방법을 개발했다. 사람 자세의 공간은 고차원에서 형성되기 때문에 무차별적인 유니폼 샘플링으로는 다루기가 어렵다. 우리는 차원의 감소와 지역적으로 계층화 된 샘플링을 이용하여 사람의 자세 공간에서 균일하거나 애플리케이션별로 편향된 분포를 생성한다. 우리의 알고리즘은 앉은 자세, 무릎 꿇은 자세,

스트레칭 및 요가와 같은 도전적인 자세를 눈에 띄게 적은 양의 학습 데이터만으로 인식하도록 했다. 또한 인식 알고리즘은 사람 자세의 특정 도메인에 대한 성능을 최대화하도록 조정될 수 있다. 우리의 알고리즘은 Kinect SDK만큼이나 똑바로 서 있는 자세도 쉽게 인식할 수 있음을 보여주면서도, 도전적인 곡예 자세 인식에 훨씬 좋은 성능을 보여준다. 둘째, 우리는 사람과 상호 작용하는 주변 물건들을 함께 인식할 수 있기를 바란다. 이를 위해 우리는 기존의 사람 자세 추정 알고리즘에 적용할 수있는 새로운 소품 인식 시스템을 제안하였고, 자세와 함께 소품 추정을 동시에 가능하게 하였다. 우리의 방법은 사람의 자세와 추가적인 물체를 동시에 처리하는 다양한 유형의 시스템에 널리 적용할 수 있다. 마지막으로 하나의 깊이 카메라 시스템을 이용하는 자세 추정 결과를 향상시켰다. 사람 자세의 모든 부분이 하나의 깊이 이미지만으로 항상 예측될 수는 없다. 어떤 경우에는 일부 신체 부위가 다른 신체 부위에 의해 가려지며, 때로는 추정 시스템이 성공하지 못할 수도 있다. 이를 보완하기 위해 우리는 사람 자세 데이터를 이용하여 오토인코더(autoencoder)라는 새로운 신경망 모델을 구성했다. 이는 방대한 규모의 자연스러운 자세 데이터로부터 생성되고, 찾는 못한 사람의 자세 관절을 새롭게 찾아낸 올바른 관절로 재구성할 수 있도록 한다. 이러한 시스템은 다양한 자세 추정 시스템에 적용되어 성능을 향상시킬 수 있다.

주요어: 컴퓨터 그래픽스, 캐릭터 애니메이션, 모션 캡쳐, 사람 자세 인식, 균일 샘플링, 기계 학습, 딥 러닝.

학번: 2008-30879